repository.ub.ac.id

IMPLEMENTASI *CONNECTED COMPONENT LABELING* UNTUK DETEKSI OBJEK PENGHALANG BAGI PENYANDANG TUNANETRA BERBASIS RASPBERRY PI

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Teknik

Disusun oleh:
Ida Yusnilawati
NIM: 145150301111027

PROGRAM STUDI TEKNIK KOMPUTER
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

UNIVERSITAS BRAWIJAYA

PENGESAHAN

IMPLEMENTASI *CONNECTED COMPONENT LABELING* UNTUK DETEKSI OBJEK PENGHALANG PENGHALANG BAGI PENYANGDANG TUNANETRA BERBASIS RASPBERRY PI

SKRIPSI

Diajukan untuk memenuhi sebagaian persyaratan
Memperoleh Sarjana Teknik

Disusun Oleh :
Ida Yusnilawati
NIM: 145150301111027

Skripsi ini telah diuji dan dinyatakan lulus pada
26 Oktober 2018
Telah diperiksa dan disetujui oleh :

Dosen Pembimbing I	Dosen Pembimbing II
Dr. Eng. Fitri Utaminingrum, S.T, M.T	Mochammad Hannats Hanafi I., S.ST., M.T.
NIP: 19820710 200812 2 001	NIK: 201405 881229 1 001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

repository.ub.ac.id

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 31 Oktober 2018

Ida Yusnilawati
NIM: 145150301111027

UNIVERSITAS BRAWIJAYA

KATA PENGANTAR

Dengan menyebut nama Allah SWT Yang Maha Pengasih dan Maha Penyayang. Puji syukur kehadirat Allah SWT karena limpahan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan skripsi dengan judul "Implementasi *Connected Component Labeling* untuk Deteksi Objek Penghalang bagi Penyandang Tunanetra Berbasis Raspberry Pi" untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Teknik.

Penulis menyadari bahwa penyusunan skripsi ini terlepas dari bantuan berbagai pihak. Oleh karena itu penulis mengucapkan terima kasih kepada pihak-pihak yang telah bersedia untuk memberikan bantuan demi kelancaran penyusunan skripsi ni diantaranya:

1. Ibu Dr. Eng. Fitri Utaminingrum, S.T, M.T selaku dosen pembimbing I yang telah dengan sabar membimbing dan mengarahkan penulis sehingga dapat menyelesaikan skripsi dengan baik.
2. Bapak Mochammad Hannats Hanafi I., S.ST., M.T. selaku dosen pembimbing II yang telah sabar membimbing dan mengarahkan penulis sehingga dapat menyelesaikan skripsi dengan baik.
3. Bapak Wayan Firdaus Mahmudy, S.Si., M.T., Ph.D. selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya Malang.
4. Bapak Tri Astoto Kurniawan, S.T., M.T., Ph.D. selaku Ketua Jurusan Teknik Informatika Universitas Brawijaya Malang.
5. Bapak Sabriansyah Rizqika Akbar, S.T., M.Eng. selaku Ketua Program Studi Teknik Komputer Universitas Brawijaya Malang.
6. Seluruh dosen FILKOM Universitas Brawijaya yang telah memberikan dukungan dan bantuan selama menempuh studi di Teknik Komputer/ Ilmu Komputer Universitas Brawijaya dan selama penyelesaian skripsi ini.
7. Kedua Orang Tua penulis serta keluarga besar atas segala do'a, nasihat, dukungan baik moril maupun materiil yang begitu besar selama menempuh masa studi hingga mendapatkan kelancaran dalam menyelesaikan skripsi ini.
8. Teman-teman "D-GENGZ" Burhan, Mila, Oggy, Linda, Yongki, Lita, Putri, Intan, Ilham, Syahriel, Adrian, Mas Tezza, Yoga dan kontrakan keramat yang selalu membantu dan memberika semangat serta motivasi selama proses pengerjaan ini.
9. Rekan-rekan Teknik Komputer 2014 yang selalu memberikan motivasi dalam menyelesaikan skripsi ini.
10. Seluruh pihak yang tidak dapat disebutkan satu persatu yang telah berperan dalam penyelesaian skripsi ini.

Pada akhirnya, penulis menyadari bahwa skripsi ini masih belum sempurna. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun.

Penulis berharap semoga skripsi ini dapat bermanfaat bagi pengembangan ilmu pengetahuan dan teknologi.

Malang, 31 Oktober 2018

Penulis
idayusnilawati@gmail.com

repository.ub.ac.id

ABSTRAK

Tunanetra adalah suatu keadaan dimana kedua indera pengelihatannya tidak berfungsi untuk menerima informasi layaknya orang awas, untuk itu membutuhkan alat bantu sehingga dapat menggantikan indera pengelihatan dalam menjalankan aktifitas sehari-hari. Namun tongkat juga masih memiliki kekurangan yaitu hanya dapat digunakan untuk meraba objek atau halangan dengan jangkauan yang terbatas. Oleh karena itu dibutuhkan penelitian untuk membantu penyandang tunanetra dalam melakukan kegiatan sehari-hari. Salah satu teknologi yang memungkinkan tunanetra dalam menjalankan aktifitas sehari hari yaitu memanfaatkan computer vision yang dapat mendeteksi sebuah objek penghalang saat Tunanetra berjalan di dalam ruangan.

Sistem ini menggunakan Raspberry Pi sebagai pemroses citra digital dan kamera webcam sebagai sensor yang dipasangkan di depan dada pengguna pada ketinggian 110cm dan kemiringan kamera sebesar 41° sehingga dapat mengambil citra di depan pengguna sampai dengan 125cm. Proses deteksi objek penghalang ini dilakukan dalam beberapa tahap yaitu meresize citra, cropping, kemudian thresholding. Proses thresholding ini memanfaatkan nilai parameter mean, median dan min-max dari citra RGB yang didapatkan dari citra RGB lantai. Selanjutnya menggunakan metode *connected component labeling* 4-*connectivity* yang pada penelitian ini digunakan untuk mendeteksi objek penghalang dari citra yang diambil dengan memberikan label pada piksel ketetanggaannya. Piksel yang telah terlabeli tersebut akan dianalisis untuk dapat mendeteksi objek penghalang. Sistem dapat mendeteksi objek penghalang pada jarak 50 cm, 75 cm, 100 cm dan 125 cm menggunakan patameter *threshold mean* memiliki akurasi sebesar 88,51%, *threshold median* sebesar 84,47% dan *threshold min-max* sebesar 66,25%. Untuk hasil akurasi sistem dalam mendeteksi objek penghalang dengan yaitu sebesar 91,66%. Hasil pengujian akurasi integrasi sistem dengan *hardware* yaitu sebesar 98,33% dan rata-rata waktu komputasi sistem yaitu sebesar 166,15 ms

Kata kunci : *tunanetra, computer vision, thresholding, connected component labeling*

UNIVERSITAS BRAWIJAYA

repository.ub.ac.id

ABSTRACT

Blind is a condition where the both senses of sight do not work to receive information like the alert person, that's why it needs auxiliary tool, so it can replace the sense of sight in carrying out daily activities. However, the stick still has a deficiency too that can only be used to touch objects or obstacles with a limited range. Therefore, research is needed to help blind people in carrying out daily activities. One of technologies that enables blind people in carrying out daily activities is to use computer vision that can detect a barrier object when a blind person walks in the room.

This system uses Raspberry Pi as a processor and a webcam camera as a censor attached in front of the user's chest at a height of 110cm and a camera tilt of 41°, so that it can take the image in front of the user up to 125cm. The detection process of this barrier object is done in several steps, such as resizing the image, cropping, then thresholding. This thresholding process utilizes the parameter values of the mean, median and min-max from the RGB image obtained from the RGB image of floor. Then using the connected component labeling 4-connectivity method in this study is used to detect barrier object from the image taken by labeling in the neighboring pixels. Pixels that have been labeled will be analyzed to be able to detect barrier object. The system can detect barrier object at a distance of 50 cm, 75 cm, 100 cm and 125 cm using the parameter of threshold mean having an accuracy of 88.51%, the threshold median is 84.47% and the threshold min-max is 66.25%. For the results of system accuracy in detecting barrier object is 91.66%. The result of study for accuracy of system integration with hardware is 98.33%, and the average time of system computing is 166.15 ms.

Keywords: *blind, computer vision, thresholding, connected component labeling*

UNIVERSITAS BRAWIJAYA

DAFTAR ISI

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

DAFTAR GAMBAR

DAFTAR TABEL

repository.ub.ac.id

BAB 1 PENDAHULUAN

1.1 Latar belakang

Tunanetra adalah suatu keadaan, dimana seseorang kehilangan pengelihatan karena kedua indera pengelihatannya tidak berfungsi untuk menerima informasi dalam kegiatan sehari-hari seperti orang awas (Harimukhti & Dewi, 2014). Pada tahun 2017, *World Health Organization* (WHO) menuliskan sebanyak 81% gangguan pengelihatan terjadi pada orang-orang berusia diatas 50 tahun keatas. Terdapat 253 juta orang di seluruh dunia mengalami tuna netra. 36 juta mengalami kebutaan dan 217 juta memiliki gangguan pengelihatan sedang hingga berat (Silvia P, 2017).

Dalam kehidupan sehari-hari penyandang tunanetra mengalami kesulitan dalam beraktifitas, salah satunya yaitu berjalan. Karena memiliki keterbatasan dalam pengeliatan kebanyakan tunanetra menggunakan tongkat sebagai pengganti indera pengelihatan dan menggunakan indera pendengaran sebagai respon dari tongkat tersebut (Al Kadafi & Utaminungrum, 2017). Namun tongkat masih memiliki kekurangan yaitu hanya dapat digunakan untuk meraba objek atau halangan dengan jangkauan yang terbatas, harus dekat dengan objek penghalang untuk diraba dan dirasakan. Untuk mengatasi masalah tersebut, berbagai kalangan dan berbagai bidang mencoba mengajukan beberapa solusi seperti penerapan teknologi sebagai pengganti indera pengelihatan.

Salah satu teknologi yang dapat dilakukan untuk mengatasi masalah tersebut adalah memanfaatkan *computer vision*. Menurut (Wahyudi & Kartowisastro, 2011), *Computer Vision* merupakan cabang ilmu Pengolahan Citra Digital yang memungkinkan komputer dapat melihat seperti manusia, sehingga dapat mengambil keputusan, melakukan aksi, dan mengenali suatu Objek.

Oleh karena itu, penulis mencoba membangun sebuah sistem yang memanfaatkan *computer vision* untuk membantu penyandang tunanetra dalam melakukan aktifitas berjalan. Pada penelitian ini, penerapan *computer vision* untuk mengidentifikasi objek berupa halangan dengan memanfaatkan keadaan lantai. Halangan yang dimaksud adalah objek seperti orang, *box*, meja atau benda lain yang berada di depan penyandang tunanetra ketika berjalan yang dapat dikategorikan sebagai halangan. Secara garis besar prinsip kerja sistem ini dapat mendeteksi objek yang dianggap sebagai penghalang menggunakan sensor kamera yang dipasang depan dada pengguna setinggi 110 cm dengan kemiringan kamera sebesar 41° sehingga dapat mengambil citra di depan pengguna samapi dengan 125 cm. Tahap awal yaitu menghitung nilai *min-max, median, mean* citra RGB pada lantai. Kemudian hasil citra RGB pada lantai akan digunakan sebagai batas *threshold*. Proses t*hresholding* pada penelitian ini digunakan untuk membedakan *background* dan *foreground.*

UNIVERSITAS BRAWIJAYA

selanjutnya yaitu mendeteksi objek menggunakan algoritma *Connected Component Labelling*.

Algoritma *Connected Component Labelling* pada penelitian ini adalah untuk deteksi objek. *Connected Component Labeling* merupakan algoritme dasar dalam pengolahan citra digital yang secara umum digunakan dalam proses yang berhubungan dengan deteksi objek (Schwenk, 2015). Penggunaan metode *Connected Component Labeling* sendiri sudah sering dipakai untuk deteksi blob/objek. Pada penelitian ini objek yang telah terdeteksi selanjutnya dianalisis untuk mendeteksi suatu halangan yang terdapat pada citra. Selanjutnya sistem akan mengeluarkan output berupa suara dari buzzer yang akan digunakan bagi penyandang tunanetra sebagai pertanda keadaan jalan yang ada di depannya.

1.2 Rumusan masalah

Berdasarkan pemaparan latar belakang tersebut, maka rumusan masalah yang bisa dikaji adalah sebagai berikut :

1. Bagaimana pengaruh *threshold mean, median* dan *min-max* pada berbagai waktu dan jarak yang berbeda-beda terhadap nilai akurasi?
2. Bagaimana akurasi sistem dalam mendeteksi *object* menggunakan *Connected Component Labelling 4-connectivy?*
3. Bagaimana rata-rata akurasi hasil integrasi sistem antara *software* dan *hardware* dalam mendeteksi objek berupa penghalang pada kondisi pagi, siang dan malam?
4. Berapa lama waktu komputasi yang dibutuhkan sistem untuk mendeteksi objek yang dianggap sebagai penghalang?

1.3 Tujuan

Adapun maksud dan tujuan dari penelitian ini adalah sebagai berikut :

1. Dapat mengimplementasikan pengolahan citra menggunakan *threshold* dari citra RGB pada berbagai waktu dan jarak yang berbeda-beda.
2. Dapat mengetahui tingkat akurasi *Connected Component Labelling* dalam mendeteksi objek berupa halangan.
3. Dapat mengetahui tingkat akurasi integrasi sistem *software* dan *hardware* dalam mendeteksi objek penghalang.
4. Mengetahui tingkat komputasi waktu dalam mendeteksi objek penghalang.

1.4 Manfaat

Dengan dilakukannya penelitian ini diharapkan dapat membantu penyandang tunanetra untuk mengetahui keberadaan objek penghalang saat berjalan.

1.5 Batasan masalah

Batasan masalah ditentukan agar permasalahan yang dirumuskan dapat lebih terfokus dan tidak meluas. Pada peneitian ini, batasan masalah tersebut antara lain :

1. Citra diambil dalam posisi *landscape*
2. Objek yang ditangkap hanya objek yang diidentifikasi berada di depan pengguna pada jarak sekitar 50 cm-125 cm
3. Diterapkan pada keadaan di dalam ruangan (*indoor)*
4. Pola lantai yang digunakan memiliki komposisi warna *cream*
5. Citra yang diambil tidak memiliki warna yang sama dengan lantai, tidak berupa kaca dan tidak terdapat bayangan/silau
6. Ukuran citra yang diproses berukuran 320 x 220 piksel
7. Perangkat ditempatkan pada ketinggian 110 cm dari lantai dengan kemiringan kamera yaitu 41°
8. Pengujian dilakukan pada pagi (197-590 lux), siang (153-965 lux) dan malam (12-79 lux)
9. Menggunakan *connected component labeling 4-connectivity*

1.6 Sistematika Pembahasan

Skematik penulisan digunakan untuk membahas penelitian ini secara garis besar. Berikut uraian dari sistematika pembahasan :

BAB I Pendahuluan

Bab ini berisi tentang latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, dan sistematika penulisan yang digunakan untuk merancang sistem pengolah citra digital untuk deteksi objek penghalang sebagai penunjuk bagi penyandang tunanetra.

BAB II Landasan Kepustakaan

Landasan kepustakaan menjelaskan tentang kajian pustaka, sumber-sumber terkait dengan dasar teori yang akan digunakan untuk menunjang berjalannya penelitian sistem deteksi objek penghalang bagi penyandang tunanetra.

BAB III Metodologi Penelitian

Membahas tentang metode dan langkah-langkah kerja yang dilakukan dalam penulisan tugas akhir. Metode yang digunakan yaitu kajian putstaka mengenai penelitian yang terkait dan kebutuhan penelitian, analisis kebutuhan, desain dan implementasi sistem, dan pengambilan kesimpulan.

BAB IV Rekayasa Kebutuhan

Membahas tentang kebutuhan yang perlu dipenuhi dalam menunjang keberhasilan sistem sesuai dengan tujuan dibuatnya sistem. Rincian dari bab ini yaitu gambaran umum sistem dan rekayasa kebutuhan

BAB V Perancangan dan Implementasi

Membahas tentang perancangan sistem baik *hardware* maupn *software* sebelum kemudian diimplementasikan menjadi sistem deteksi objek penghalang

bagi penyandang tunanetra. Implementasi pada bab ini berupa implementasi pengolahan citra pada video hingga mendapatkan keluaran.

BAB VI Pengujian dan Analisis

Membahas tentang cara pengujian sistem yang dilakukan secara bertahap maupun keseluruhan sistem. Analisis terhadap sistem yang dilakukan detelah tahap pengujian. Pengujian dan analisis sitem dilakukan pada *hardware*. Deteksi objek penghalang dan sistem secara keseluruhan.

BAB VII Penutup

Memuat kesimpulan dan saran yang dihasilkan dari keseluruhan proses pengerjaan penelitian sistem deteksi objek penghalang bagi penyandang tunanetra.

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Penelitian pertama dilakukan oleh (Ardhianto, et al., 2013) yang berjudul "Implementasi Metode Image *Substracting* dan Metode *Regionprops* untuk Mendeteksi Jumlah Objek Berwarna RGB pada file *Video*" Inti penelitian ini yaitu untuk mendeteksi jumlah objek berwarna RGB pada file *video*. Pada penelitian tersebut mereka menggunakan *Connected Component Labeling* pada tahap *pre*-processsing. Algoritma *Connected Component Labeling* digunakan untuk melabeli tiap objek pada gambar binary dengan suatu label unik dan mengelompokkan piksel sebagai satu objek.

Penelitian kedua dilakukan oleh (Rizki, et al., 2010) yang berjudul "*Connected Component Analysis* Sebagai Metode Pencarian Karakter Plat dalam Sistem Pengenalan Plat Nomor Kendaraan". Pada penelitian tersebut mereka menggunakan *Connected Component Analysis* untuk melakukan proses pencarian plat nomor kendaraan, yang mana karakter pada citra disegmentasi tanpa harus melewati proses pencarian lokasi platnya terlebih dahulu. Rizki, et al juga menjelaskan bahwa *connected component labeling* adalah proses memberikan label pada *binary image* untuk memisahkan tiap karakter. Pada penelitian tersbut objek akan diberi label yang sama, sedangkan piksel *background* tidak diberikan label dengan tujuan untuk memisahkan karakter pada citra. *Selanjutnya* untuk proses pengenalan karakter dalam plat digunakan metode *Neural Network*. Dengan metode ini, karakter plat kendaraan tetap dapat dikenali meskipun warna dasar plat sama dengan badan kendaraan.

Pada penelitian yang dilakukan oleh (Putranto, et al., 2012) yang berjudul "Segmentasi Warna Citra dengan Deteksi Warna HSV untuk Mendeteksi Objek". Pada penelitian tersebut memaparkan penerapan metode segmentasi warna dengan deteksi warna HSV untuk menghasilkan objek segmen citra berupa *blob* sehingga dapat terdeteksi komputer. *Blob* pada penelitian ini digunakan untuk pemetaaan dan perhitungan objek. Proses perhitungan *blob* dilakukan dengan melakukan analisis piksel yang bertetangga yang memanfaatkan relasi piksel 8-neighbours. Letak dan label piksel yang bertetangga tersebut digunakan untuk membentuk suatu *blob*.

2.2 Dasar Teori

2.2.1 Tunanetra

Seseorang dikatakan tunanetra apabila ketajaman pengelihatannya kurang dari 6/21 meter. Artinya berdasarkan tes seseorang tersebut hanya mampu membaca huruf pada jarak 6 meter tetapi pada orang awas dapat dibaca pada jarak 21 meter (Wardani, 2012). (Isharwati, 2008) Juga menjelaskan bahwa tunanetra merupakan suatu kedaaan dimana mata seseorang memiliki kekurangan daya dalam melihat sehingga menggunakan alat batu atau

menggunakan panca indera yang lain seperti pendengaran. Dari uraian diatas, pengertian tunanetra adalah suatu keadaaan dimana fungsi panca indera pengelihatannya tidak berfungsi sebagai penerima informasi seperti halnya orang awas (Harimukhti & Dewi, 2014).

Tunanetra dikategorikan menjadi dua, yaitu buta (*totally blind)* dan pengelihatan lemah (*low vision)* (Sardegna, 2002). Buta adalah suatu keadaan dimana seseorang tidak bisa menerima cahaya dari luar sama sekali, sedangkan pengelihatan lemah yaitu seseorang yang masih mampu menerima cahaya dari luar namun ketajamannya lebih dari 6/21 meter, atau orang yang mampu membaca *headline* pada surat kabar (Soemantri, 2012).

Secara ilmiah, tunanetra disebabkan oleh berbagai faktor yaitu faktor dari dalam diri *(internal)* ataupun faktor dari luar *(external)*. Faktor-faktor *internal* tersebut dapat berupa saat keadaan masih bayi atau selama dalam kandungan. Hal ini kemungkinan karena faktor gen, kondisi psikis ibu, kekurangan gizi, keracunan obat dan sebagainya. Faktor-faktor *external* yaitu kondisi ketika sudah dilahirkan, contohnya terkena penyakit, kecelakaan, pengaruh alat bantu medis saat melahtkan atau kurang gizi, vitamin dan sebagainya (Soemantri, 2012).

Tunanetra disebabkan oleh 2 faktor yaitu faktor dalam diri (*internal*) atau faktor luar (*external*). Hal-hal yang termasuk faktor internal yaitu faktor-faktor yang erat hubungannya dengan keadaan bayi selama masih di dalam kandungan. Sedangkan hal-hal yang termasuk faktor eksternal yaitu faktor yang terjadi setelah bayi dilahirkan meliputi kecelakaan, kurang gizi atau vitamin, peradangan mata karena penyakit, bakteri maupun virus (Wardani, 2012).

2.2.2 *Computer Vision*

Computer vision adalah Ilmu pengetahuan yang mempelajari bagaimana komputer dapat mengenali objek yang diamati atau diobservasi, dapat menampilkan objek dan mengkoleksi data secara visual komputer (Helmirawan, 2012). Irianto juga menjelaskan bahwa *computer vision* merupakan suatu pembelajaran menganalisa gambar atau video guna memperoleh hasil yang sebagaimana manusia lakukan (Irianto, 2010).

Tujuan utama *computer vision* adalah agar komputer meniru kemampuan perseptual mata dan otak manusia. Manusia melihat objek dengan diteruskan ke otak untuk diinterprestasi sehingga dapat mengambil keputusan (Irianto, 2010).

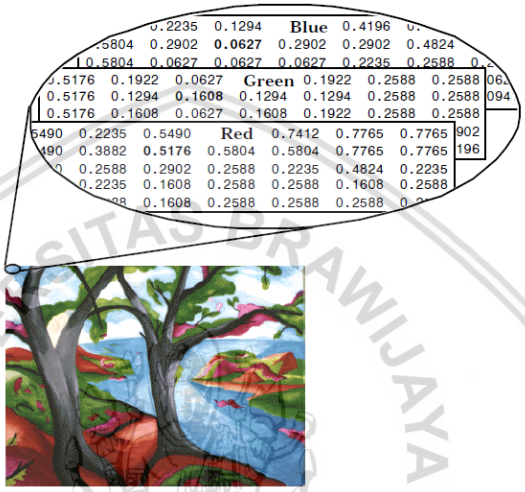
2.2.3 *Digital Image Processing*

Digital Image Processing atau pengolahan citra digital yaitu sebuah disiplin ilmu yang mempelajari mengenai teknik-teknik pengolahan citra. Citra yang dimaksud yaitu gambar diam (foto) maupun gambar bergerak *(video)* (Kusumanto & Tompunu, 2011). Pengolahan citra digital menggunakan teknologi *computer vision* yang saat ini telah banyak digunakan sebagai objek penelitian salah satunya yaitu pengolahan berdasarkan warna (Kusumanto, et al., 2011).

Kusumanto menjelaskan bahwa pada aplikasi pengolahan citra digital, jenis citra digital dibagi menjadi 3, *color image, black and white image* dan *binary image.*

1. *Color image* atau RGB *(red, green, blue)*

   Pada *color image* masing-masing piksel pada citra digital memiliki warna tertentu, warna-warna tersebut diantaranya merah (*red),* hijau *(green),* dan buru (*blue)*. Jika masing-masing memiliki range 0 – 255 maka totalnya yaitu 255^3 = 16.581.375 (16K). *Color image* terdiri dari tiga matriks yang mewakili nilai merah, hijau dan biru pada setiap pikselnya.

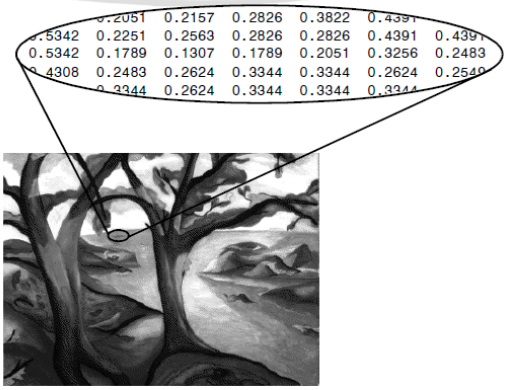


Gambar 2. 1 *Color image* atau RGB

Sumber : (Robotix, 2018)

2. *Black and White (grayscale)*

   Pada citra digital *black and white (greyscale),* setiap pikselnya memiliki gradasi warna putih hingga hitam. Setiap piksel dapat diwakili oleh 8 bit atau 1 byte. Salah satu fungsi citra digital *black and white* yaitu digunakan dalam dunia kedokteran untuk X-ray.

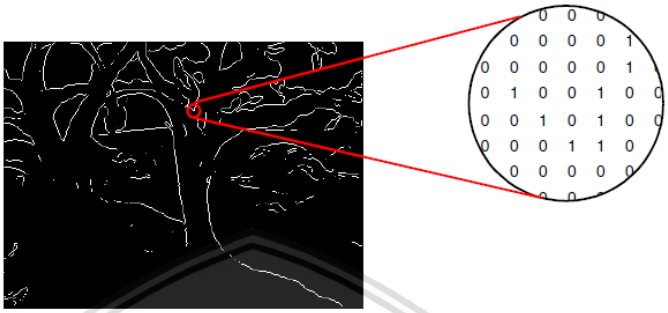


Gambar 2. 2 *Greyscale image*

Sumber : (Robotix, 2018)

3. *Binary Image*

   Setiap piksel pada citra digital *binary image*, hanya terdiri atas warna hitam dan putih karena hanya ada dua warna untuk setiap piksel, maka hanya perlu 1 bit per piksel (0 atau 1). Gambar yang direpresentasikan dengan biner sangat cocok untuk teks, sidik jari *(figner print)* atau gambar arsitektur.

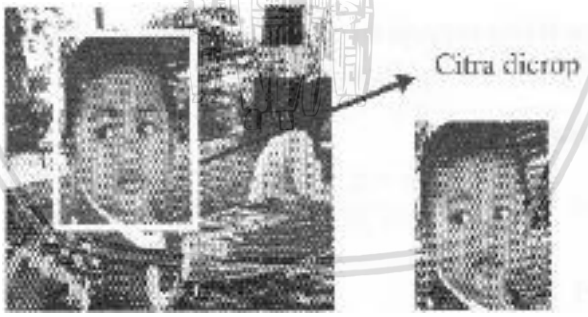


Gambar 2. 3 *Binary Image*

Sumber : (Robotix, 2018)

2.2.4 Cropping Image

Cropping image atau pemotongan citra yang bertujuan untuk membuat image atau citra baru dengan mengambil elemen citra yang diinginkan pada citra digital (Budisanjaya & Kumara, 2013) Pada penelitian ini *cropping image* digunakan untuk mengambil citra dengan ukuran 150 x 150 piksel pada citra asli. Berikut contoh pemotongan citra W x H pada Gambar 2.4



(a) Citra asli (b) Citra hasil *cropping*

Gambar 2. 4 Contoh *cropping* citra

Sumber : (Budisanjaya & Kumara, 2013)

2.2.5 *Image Resizing*

Image resizing yaitu merubah suatu ukuran gambar menjadi lebih kecil dari ukuran sebenarnya. Pada penelitian ini image resizing digunakan untuk merubah ukuran citra menjadi lebih kecil yaitu 320 x 220 piksel. Hal ini dilakukan agar komputasi citra tidak terlalu berat.

2.2.6 Segmentasi Citra dan *Thresholding*

Segmenasi merupakan langkah utama dan langkah yang penting dalam suatu pengenalan objek *(object recognation)*. Proses segmentasi ini merupakan proses untuk memisahkan objek satu dengan objek yang lain. Masing-masing objek pada proses segmentasi, citra dapat diambil secara terpisah sehingga dapat digunakan sebagai masukan proses yang lain. Segmentasi merupakan suatu proses pembagian citra ke dalam wilayah *(region)* yang memiliki kesamaan fitur, contohnya tingkat keabuan *(greyscale),* tekstur *(texture)*, gerakan *(motion)* (Murinto & Harjoko, 2009). Teknik yang sering digunakan untuk memisahkan antara objek *background* dan *foreground* adalah binerisasi atau *thresholing* (Fanani, et al., 2012)

Metode *thresholding* adalah salah satu teknik pengolahan citra digital yang digunakan untuk mengubah citra dengan format *true color* menjadi citra biner *(binary image)* yang hanya memiliki 2 buah bilai (0 atau 1). Konversi citra hitam-putih ke citra biner dilakukan dengan operasi pengambangan (Thresholding). Operasi ini mengelompokkan nilai derajad keabuan setiap piksel ke dalam dua kelas, hitam dan putih. Pada citra hitam-putih terdapat 256 level, artinya mempunyai skala dari "0" sampai "256" atau [0,255]. Dalam hal ini nilai intensitas 0 menyatakan hitam, dan nilai intensitas 255 menyatakan putih, dan nilai antara 0 sampai 255 menyatakan warna keabuan yang terletak antara hitam dan putih (Firdausy, et al., 2007).

Thresholding dapat dikategorikan menjadi 2, yaitu *local thresholding* dan *global thresholding. Local Thresholding* setiap piksel dihitung dengan nilai *threshold* berbeda berdasarkan data statistik atau *range,* variansi atau keadaaan piksel tetangganya. Penggunaan *local thresholding* bergantung pada keadaan seperti *region,* karakteristik citra dan waktu eksekusi. Sedangkan *global thresholding* dapat diterapkan ketika intensitas antara *background* dan *foreground* memiliki perbedaan yang tinggi (N & S, 2016). Salah satu contoh penggunaan *global threshold* adalah *color filtering*. *Color flltering* dilakukan dengan menyaring suatu nilai warna tertentu menggunakan 2 batas yaitu minimal dan maksimal (Amri, et al., 2014).

Thresholding pada penelitian ini menggunakan *local thresholding color filtering* yang digunakan untuk membedakan *background* dan *foreground* dengan memanfaatkan model warna RGB. Setiap komponen warna RGB memiliki nilai *threshold* masing-masing, seperti pada persamaan 2.1

$$g(x,y) = \begin{cases} 1, (B > Bmin) \quad (B < BMax) \\ \quad (G > Gmin) \quad (G < GMax) \\ \quad (R > Rmin) \quad (R < RMax) \\ \\ 0, else \end{cases} \qquad (2.1)$$

Menurut (Pramana, 2015) terdapat berbagai metode dalam memilih *threshold.* Metode paling sederhana dilakukan dengan menghitung Nilai *threshold*

dengan cara menghitung nilai *threhsold mean* (rata-rata), *threshold median* (nilai tengah) atau *min-max* dari citra.

2.2.6.1 Mean

Mean yaitu nilai rata-rata yang pada penelitian ini *threshold mean* didapatkan dari rata-rata nilai citra RGB. Citra RGB yang dimaksudkan yaitu data latih lantai. Pengambilan data latih ini dilakukan sebanyak 30 kali, yaitu pada saat pagi diambil 10 kali citra lantai, siang diambil 10 kali citra lantai dan malam diambil 10 citra lantai. Perhitungan nilai *threhold mean* dari data citra sebagaimana pada persamaan 2.2

$$mean = \frac{1}{n} \sum_{i=1}^{n} x_1 \quad (2.2)$$

Keterangan
n : Jumlah data
x : nilai data ke i

2.2.6.2 Median

Median yaitu nilai tengah yang pada penelitian ini *threshold median* didapatkan dari nilai tengah data citra RGB lantai. Perhitungan nilai *threhold median* dari data citra sebagaimana pada persamaan 2.3

$$Median = \frac{1}{2}\left(x_{\left(\frac{n}{2}\right)} + x_{\left(\frac{n}{2}+1\right)}\right) \quad (2.3)$$

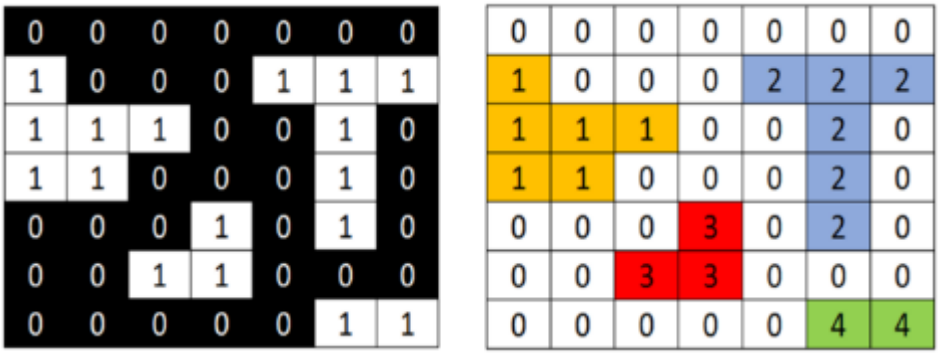
Keterangan
n : Jumlah data
x : nilai data

2.2.6.3 Threshild Min-Max

Threshold Min-Max yaitu batas threshold minimal dan maksimal yang pada penelitian ini *threshold min-max* didapatkan dari nilai minimal data citra RGB lantai dan nilai maksimal data citra RGB lantai.

2.2.7 Connected Component Labeling

Connected Component Labeling digunakan pada *computer vision* untuk mengklasifikasikan region atau objek pada citra digital. *Connected component labeling* umumnya digunakan pada *binary image* dari proses *thresholding*. Algoritma ini dapat menghitung, memfilter atau *tracking* (Salem, 2014). Hendry juga menjelaskan bahwa algoritma ini memanfaatkan teori *connectivity* piksel pada citra. Piksel antar piksel dikatakan *connected* (terkoneksi) jika mematuki aturan ketetanggan piksel. Dengan demikian piksel-piksel dikatakan bertetangga *(neighbourhood)* pada dasarnya memiliki konektifitas *(connectivity)* satu sama lain (Hendry, 2010). Metode ini dapat melabeli tiap region yang memiliki koneksi dengan label yang unik. Jumlah objek yang terdapat dalam suatu citra biner juga dapat diketahui dengan melihat jumlah region yang didapatkan.



(a) (b)

Gambar 2. 5 Contoh citra biner yang dilabeli

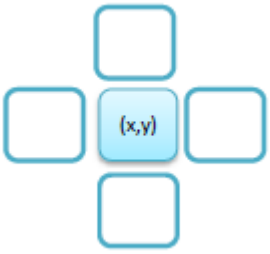
(b) Citra biner ; (b) citra yang telah dilabeli dengan metode *connected component labeling*

Sumber : (Mirul, 2017)

Metode pelabelan ini dilakukan dengan mengecek nilai piksel pertama yaitu baris pertama, kolom pertama. Jika nilai tersebut 0 *(background)* maka dilewati dan berpindah pada kolom selanjutnya pada baris yang sama. Jika nilai tersebut 1 maka dicek piksel tetangganya. Apabila piksel tetangga terdekatnya belum terlabeli, beri label pada piksel yang tersebut dengan label 1. Apabila piksel tetangga terdekatnya telah terlabeli, beri label yang sama dengan label teranggagga. Apabila memiliki lebih dari satu tetangga terdekat dengan label berbeda, beri label menggunakan label terkecil dari tetangga-tetangga tersebut. Setiap selesai melakukan pelabelan dan bertemu kembali dengan *background*, perbarui indeks label. Proses ini dilakukan sampai keseluruhan baris dan kolom dicek. Hasil pelabelan tersebut dipegaruhi pula oleh tipe konektifitas yang digunakan. Terdapat 2 tipe *connectivity* yang sering digunakan untuk pengecekan label ketetanggaan, yaitu dengan 4-*connectivity* dan 8-*connectivity*.

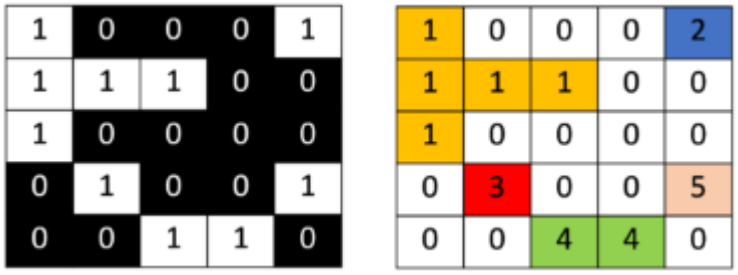
a. 4-*connectivity*

Tipe ketetanggaan pada tipe 4-*connectivity* ini hanya mengecek *piksel* yang ada pada bagian atas, bawah, kanan dan kiri dari piksel pusat.



Gambar 2. 6 *Connected Component Labeling* tipe 4-*connectivity*

Sumber : (Al Kadafi, 2017)



(a) (b)

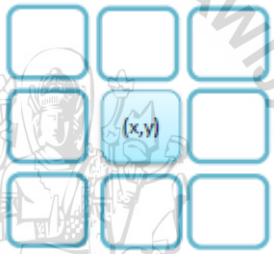
Gambar 2. 7 Contoh citra biner yang telah terlabeli dengan tipe 4-*connectivity*

(a) Citra biner ; (b) citra yang telah dilabeli dengan metode *connected component labeling* tipe 4-*connectivity*

Sumber : (Mirul, 2017)

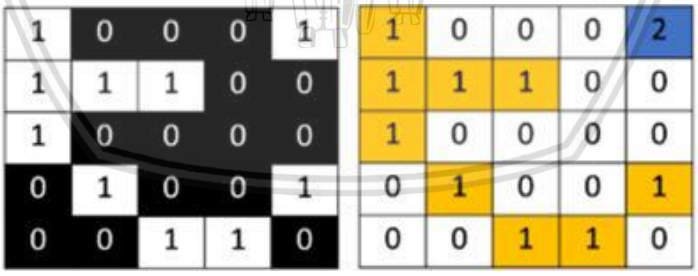
b. 8-*connectivity*

Tipe ketetanggaan pada tipe 8-*connectivity* ini mengecek *piksel* secara *horizontal, vertical* dan diagonal di sekitarnya.



Gambar 2. 8 *Connected component labeling* dengan tipe 8-*connectivity*

Sumber : (Al Kadafi, 2017)



(a) (b)

Gambar 2. 9 Contoh citra biner yang telah terlabeli dengan tipe 8-*connectivity*

(a) Citra biner ; (b) citra yang telah dilabeli dengan metode *connected component labeling* tipe 8-*connectivity*

Sumber : (Mirul, 2017)

2.2.8 Raspberry Pi 3

Raspberry Pi adalah modul *micro computer* yang memiliki *port* digital *input output* seperti pada board *microcontroller*. Rasberry Pi dilengkapi dengan port/koneksi untuk *display* monitor PC atau berupa TV, serta koneksi USB untuk

keyboard dan *mouse.* Operating System (OS) yang dibutuhkan untuk menggunakan Raspberry yaitu bermacam-macam seperti windows, linux, mac , Unix dan sebagainya. Raspberry Pi 3 sudah memiliki modul WiFi dan *bluetooth* sehingga dapat memudahkan untuk melakukan komunikasi (MATT, 2016)

Adapun spesifikasi dari Raspberry pi 3 :

Tabel 2. 1 Spesifikasi Raspberry Pi

No.	Spesifikasi	Keterangan
1.	Prosesor	Quad Core 1.2GHz 64 bit CPU
2.	SoC	*Broadcom* BCM2837
3.	RAM	1GB LPDDR (900 MHz)
4.	WiFi	802.11n
7.	Bluetooth	Bluetooth v4.1 *Classic (*Low Energy)
8.	GPIO	40pin *header, populated*
9.	Ports	4x USB 2.0, *Camera Serial Interfave* (CSI) port, *Display Serial Interface* (DSI) port, MicroSD card slot, HDMI

Gambar 2. 10 Raspberry Pi

sumber : (Raspberry Pi Foundation, 2016)

2.2.9 Kamera Logitech C310

Kamera Logitech C310 merupakan salah satu kamera produksi dari Logitech. Pada penelitian ini kamera logitech C310 digunakan untuk mengambil citra digital. Adapun spesifikasi dari kamera Logitech C310 ini yaitu memiliki konektor USB 2.0 sehingga dapat digunakan pada Raspberry Pi, memiliki resolusi sebesar 1280 x 720 *piksel* (Kadmiri, et al., 2012). Kamera ini dapat meng*capture* video dengan ukuran mencapai 730p dan memiliki *frame rate max* yaitu 30fps pada resolusi 640x480 (Logitech, 2018)

Gambar 2. 11 Kamera Logitech C310

Sumber : (Logitech, 2018)

2.2.10 *Buzzer*

Buzzer merupakan komponen elektronika yang berfungsi untuk mengubah getaran listrik mejadi getaran suara. Prinsip kerja *buzzer* hampir sama halnya dengan *loud speaker*. *Buzzer* terdiri dari kumparan yang telah terpasang diafragma kemudian kumparan tersebut dialiri arus sehingga menjadi elektromagnet. Kumparan tadi akan tertarik ke dalam atau keluar, tergantung dari arah arus dan polaritas magnetnya, karena kumparan dipasang pada diafragma maka setiap gerakan kumparan akan menggerakkan diafragma secara bolak-balik sehingga membuat udara bergetar yang akan menghasilkan suara (Sulistiyowati & Febriantoro, 2012).

Pada penelitian ini, *buzzer* digunakan untuk keluaran dari sistem yaitu berupa *buzzer* berbunyi jika keadaan/sistem mendeteksi objek di depan pengguna. Adapun spesifikasi *buzzer* yang dapat dilihat pada tabel 2.2

Tabel 2. 2 Spesifikasi *Buzzer*

No	Spesifikasi	Keterangan
1	Tegangan	6V DC
2	Tegangan Operasional	4 to 8V DC
3	*Rated Current**	≤30mA
4	Keluaran suara pada 10cm*	≥85dB
5	Frekuensi Resonansi	2300 ±300Hz
6	Bunyi	Berlanjut
7	Temperatur Operasional	-25°C to +80°C
8	Temperatur Penyimpanan	-30°C to +85°C

Gambar 2. 12 *Buzzer*

Sumber : (HobbyTronics, 2018)

2.2.11 OpenCV

OpenCV merupakan *library* yang terdapat fungsi-fungsi pemrograman untuk teknologi *computer vision* secara *real time* (Cahyana, et al., 2014). Antarmuka bahasa yang digunakan oleh OpenCV yaitu bahasa C++, Python dan java. Sistem Operasi yang dapat mendukung OpenCV diantaranya yaitu windows, Mac Os, Linux, Android dan IOS (OpenCV, 2018)

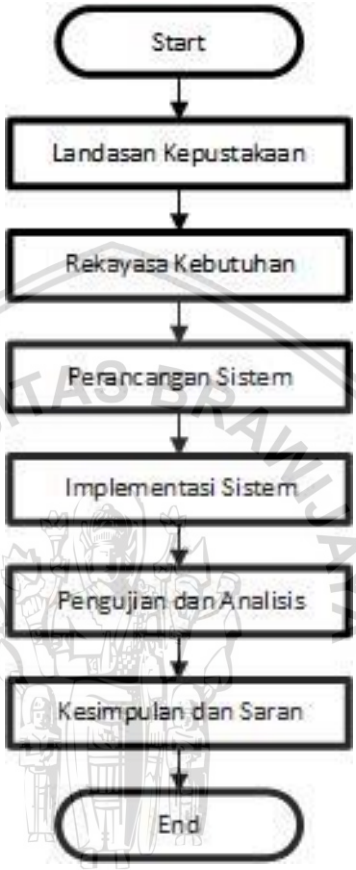
Gambar 2. 13 Logo *software* OpenCV

Sumber : (OpenCV, 2018)

BAB 3 METODOLOGI

3.1 Alur Metodologi Penelitian

Alur metodologi penelitian yang dilakukan secara umum dapat dilihat dari Gambar 3.1 berikut ini :



Gambar 3. 1 Diagram Alir Metodologi Penelitian

3.2 Landasan Kepustakaan

Pada landasan kepustakaan ini dilakukan untuk mendapatkan kajian pustaka dan dasar teori sebagai acuan dalam penelitian ini. Kajian pustaka meliputi jurnal pendukung yang didapatkan dari penelitian-penelitian sebelumnya. Dasar teori yang digunakan sebagai referensi dalam penelitian ini yaitu :

1. Tunanetra
2. *Computer Vision*
3. *Digital Image Processing*
4. Segmentasi Citra
5. *Connected Component Labeling*
6. Raspberry Pi 3
7. Kamera Logitech C310
8. *Buzzer*
9. OpenCV

3.3 Rekayasa Kebutuhan

Rekayasa kebutuhan bertujuan untuk menjelaskan kebutuhan sistem yang akan dibangun. Rekayasa kebutuhan dilakukan dengan mengidentifikasi kebutuhan penelitian secara keseluruhan. Kebutuhan sistem dibagi menjadi dua yaitu, kebutuhan perangkat lunak dan kebutuhan perangkat keras.

3.3.1 Kebutuhan Perangkat Keras

Kebutuhan perangkat keras yang dibutuhkan dalam membangun sistem adalah sebagai berikut :

1. Laptop
2. Kamera webcam Logitech C310
3. Raspberry Pi 3
4. *Buzzer*

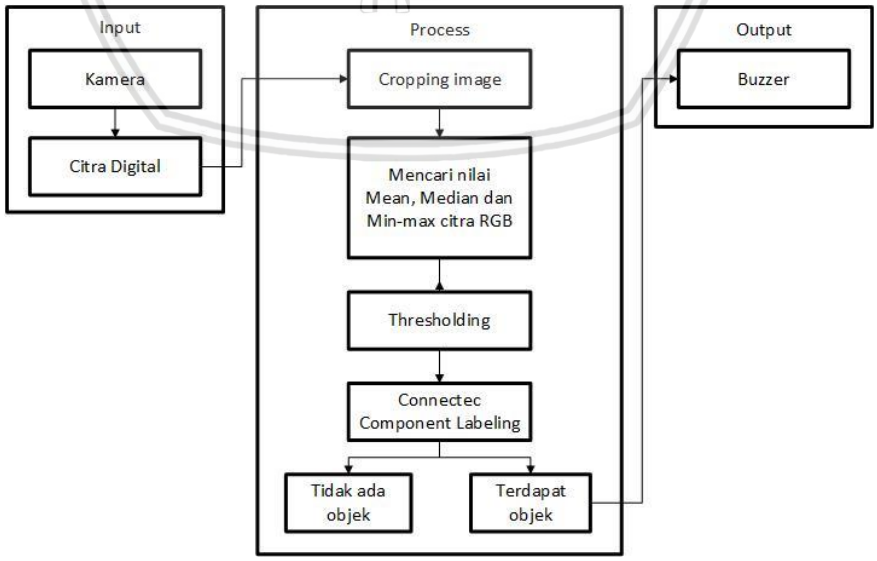
3.3.2 Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak yang dibutuhkan dalam membangun sistem adalah sebagai berikut :

1. *Operation System* (OS) Windows 7
2. VNC *Viewer*
3. *Library* OpenCV

3.4 Perancangan Sistem

Perancangan sistem merupakan tahap untuk menggambarkan bentuk sistem secara umum. Perancangan menghubungkan antara kebutuhan dan implementasi dengan menekankan pada cara sistem memenuhi kebutuhan. Rancangan umum arsitektur sistem dapat dilihat pada blok diagram pada Gambar 3.2.



Gambar 3. 2 Blok Diagram Arsitektur Umum Sistem

Pada Gambar 3.2 dapat dilihat bahwa *input* sistem berasal dari kamera. Kamera akan menangkap citra digital *video* dan mengirimkannya ke Raspberry Pi untuk dilakukan pemrosesan. Langkah pertama, video akan dipotong *(cropping)* pada bagian bawah dengan ukuran sebesar 150 x 150 piksel. Hal ini dilakukan guna mempercepat waktu komputasi dan meminimalisir kesalahan dalam mendeteksi. Selanjutnya, citra digital yang sudah melalui proses *cropping* kemudian dihitung nilai *mean, median* dan *min-max*nya untuk dijadikan sebagai batas pada proses *thresholding*. Pada proses *thresholding,* terjadi pemisahan *backdground* dan *foreground* yang kemudian dilanjutkan dengan deteksi objek menggunakan *connected component labeling.* Sistem akan memberikan output berupa suara melalui *buzzer* ketika sistem berhasil mendeteksi objek yang ada di depan pengguna.

3.5 Implementasi Sistem

Implementasi sistem dilakukan dengan mengacu pada perancangan sistem. Implementasi dilakukan dengan menggunakan bahasa pemrograman phyton menggunakan aplikasi OpenCV.

Tujuan utama dari implementasi adalah agar sistem dapat digunakan untuk mendeteksi objek yang dianggap sebagai penghalang. Ketika sistem dijalankan, secara otomatis akan melakukan pendeteksian area di sekitar pengguna. Ketika sistem mendeteksi objek yang dianggap sebagai halangan maka sistem akan mengeluarkan *output* berupa suara alarm.

3.6 Pengujian dan Analisis

Pada tahap pengujian dan analisis dilakukan untuk mengetahui keberhasilan dan akurasi sistem yang telah dibangun pada tahap implementasi. Parameter keberhasilan yang dimaksudkan adalah sebagai berikut :

1. Pengujian pengaruh nilai *threshold* pada waktu dan jarak yang berbeda-beda
2. Pengujian akurasi sistem dalam mendeteksi objek menggunakan *connected component labelling*
3. Pengujian akurasi *hardware* dan *software* sistem dalam deteksi objek berupa halangan pada berbagai waktu
4. Pengujian waktu komputasi sistem

3.7 Pengambilan Keputusan

Pada tahap pengambilan kesimpulan dilakukan setelah melakukan semua tahap perancangan, implementasi dan pengujian sistem dilakukan. Kesimpulan diambil untuk menjawab rumusan masalah yang telah ditetapkan sebelumnya. Tahap akhir dari penulisan adalah menentukan saran dan untuk memberikan informasi bagi pengembangan sistem selanjutnya agar dapat memperbaiki kesalahan atau kekurangan pada penelitian ini.

BAB 4 REKAYASA KEBUTUHAN

4.1 Gambaran Umum Sistem

Pada penelitian ini, sistem yang dibangun adalah sistem untuk mendeteksi objek dari citra hasil tangkapan kamera yang diklasifikasikan sebagai objek penghalang. Kamera diletakkan dengan tinggi 110 cm dan menghadap kebawah dengan kemiringan 41°. Jarak terpendek yang dapat terlihat oleh kamera yaitu 50 cm dan jarak terjauh yang dapat terlihat oleh kamera yaitu 125 cm. Hasil citra yang ditangkap oleh kamera selanjutnya akan dipotong (*cropping)* pada bagian bawah dengan ukuran 150 x 150 piksel dan setiap piksel RGB pada hasil cropping tersebut akan dihitung nilai *mean, median* dan *min-max*nya. Kemudian dari hasil cropping tersebut akan dijadikan sebagai input pada proses segmentasi. Proses segmentasi ini menggunakan batas *threshold* yang didapatkan dari perhitungan RGB *mean, median* dan *min-max*nya untuk membedakan *foreground* dan background. Pada sistem ini juga menggunakan metode *Connected Component Labelling* untuk mendeteksi objek penghalang setelah dilakukannya proses *thresholding.* Setelah objek teridentifikasi, sistem akan mengambil keputusan apakah objek tersebut merupakan suatu halangan atau bukan.

4.1.1 Tujuan

Tujuan dari dibuatnya sistem ini yaitu dapat mendeteksi objek penghalang bagi penyandang tunanetra menggunakan sensor kamera yang dipasang pada dada depan pengguna dan dapat memberikan keluaran berupa bunyi dari *buzzer* sebagai penanda terdapatnya objek penghalang atau tidak.

4.1.2 Karakteristik Pengguna

Karakteristik pengguna yang dapat menggunakan sistem ini adalah penyandang tunanetra dengan ketinggian melebihi 110 cm dengan postur tubuh normal dan tegak. Sehingga pengguna dapat menggunakan sistem dengan maksimal.

4.1.3 Lingkup Operasi

Pada implementasi sistem ini terfokus pada mendeteksi objek penghalang. Oleh karena itu sistem harus dirancang semaksimal mungkin agar dapat mendeteksi objek penghalang dengan akurat. Berikut adalah persyaratan lingkungan operasi sistem yang harus dilakukan :

1. Sensor kamera menggunakan *webcam* logitech C310 yang dipasangkan pada dada depan pengguna dengan ketinggian 110 cm dan kemiringan kamera yaitu 41°
2. Sistem digunakan di dalam ruangan dengan kondisi lantai yaitu satu warna berupa warna krem atau putih dan tidak terdapat bayangan. Pencahayaan yang cukup yaitu dengan rentang 197 – 590 lux pada konsisi pagi, siang 153-965 lux dan malam 12-79 lux.

4.1.4 Batasan Perancangan dan Implementasi

Dalam pembuatan sistem deteksi objek penghalang bagi penyandang tunanetra terdapat batasan-batasan. Batasan perancagnan dan implementasi pada sistem ini yaitu sebagai berikut :

1. Sistem digunakan untuk di dalam ruangan.
2. Warna lantai satu warna yaitu warna *cream.*
3. Sistem harus digunakan pada depan dada pengguna dengan ketinggian 110 cm dan kemiringan kamera 41° sehingga dapat menyorot keadaan lantai 50 cm didepan pengguna hingga 125 cm didepan pengguna.
4. Kondisi pencahayaan yaitu pada pagi hari dengan rentang 197 – 590 lux, siang 153-965 lux dan malam 12-79 lux.
5. Kondisi lantai tidak terdapat bayangan/silau.

4.1.5 Asumsi dan Ketergantungan

Asumsi dan ketergantungan pada sistem ini antara lain sebagai berikut :

1. Pin sensor kamera dan *buzzer* dihubungkan ke Raspberry Pi sesuai *datasheet* yang dianjurkan.
2. Untuk mendapatkan citra digital yang jelas dari kamera maka diperlukan kondisi pencahayaan ruangan yang tidak terlalu gelap.
3. Untuk mendeteksi objek penghalang, pengguna harus berdiri tegak.

4.2 Analisis Kebutuhan Sistem

Analisis kebutuhan sistem dilakukan untuk menggali semua kebutuhan yang diperlukan untuk sistem ini. Dalam melakukan analisis kebutuhan sistem terdiri atas beberapa kebutuhan yang perlu dijabarkan yakni kebutuhan fungsional dan kebutuhan non fungsional, dimana kebutuhan non fungsional terdiri dari kebutuhan perangkat keras dan kebutuhan perangkat lunak.

4.2.1 Kebutuhan Fungsional

Berikut ini adalah kebutuhan fungsional yang harus mampu dilakukan oleh sistem :

1. Sensor kamera dapat mengambil citra berupa video

    a. Penjelasan dan Prioritas

    Sensor kamera mengambil masukan dengan membaca citra digital video dari kamera *webcam* yang diletakkan pada dada depan pengguna setinggi 110 cm dengan kemiringan kamera yaitu 41°, sehingga kamera dapat menyorot keadaan lantai dengan jarak terpendek 50 cm dari pengguna dan jarak terjauh yaitu 125 cm dari pengguna. Citra video dikirimkan oleh kamera ke Raspberry Pi melalui USB. Fungsi ini memiliki prioritas tinggi dan menjadi sumber data input untuk di proses pada Raspberry Pi

b. Respon Sistem

Ketika kamera dihubungkan dengan Raspberry Pi dengan kabel USB, maka kamera dapat mulai beroperasi dan dapat mengambil citra untuk mendapatkan *input* yang dibutuhkan. Jika *input* berhasil didapatkan maka sistem selanjutnya dapat memprose pengolahan citra digital pada Raspberry Pi

2. Razpberry Pi dapat melakukan proses citra digital dari sensor kamera

a. Penjelasan dan Prioritas

Sistem mendapatkan data masukan dari kamera berupa citra. Kemudian data tersebut akan di proses oleh Raspberry Pi. Sistem akan menghitung nilai *mean, median,* dan *min-max* piksel BGR dan menentukan batas-batas *threshold.* Selanjutnya sistem juga akan melakukan *thresholding* kemudian mendeteksi ada atau tidaknya objek penghalang menggunakan *connected component labeling.* Prioritas fungsi ini tinggi karena guna menentukan nilai keluaran dari sistem.

b. Respon Sistem

Setelah mendapatkan masukan berupa citra digital, sistem melakukan proses pemotongan *(cropping)* pada bagian bawah video sesuai dengan ukuran 150 x 150 piksel. Selanjutnya, citra BGR yang sudah melalui proses *cropping* akan dicari nilai *mean, median* dan *min-max*nya. Hasil dari perhitungan tersebut akan dijadikan batas *threshold* untuk proses *thresholding.* Pada proses *thresholding,* sistem membedakan *background* dan *foreground* menggunakan 3 batas *threshold* yang berbeda yaitu, nilai *threshold mean, threshold median* dan *threshold min-max*. Setelah proses tersebut akan dilakukan proses deteksi objek menggunakan *connected component labeling*.

3. Hasil keluaran dari sistem setelah pemrosesan Raspberry Pi yaitu bunyi dari Buzzer

a. Penjelasan dan Prioritas

Fungsi ini digunakan untuk memberikan keluaran dari sistem berupa audio. Setelah melalui tahap deteksi objek penghalang, kemudian sistem akan memicu *buzzer* untuk bekerja. Terdapat 2 bunyi sebagai keluaran, yaitu bunyi *buzzer* keras dan lemah.

b. Respon Sistem

Sistem dapat mengeluarkan bunyi buzzer keras dan kecil sesuai kondisi objek penghalang dan posisi pengguna. Ketika objek penghalang berada pada area 1 di depan pengguna maka *buzzer* akan berbunyi keras, sedangkan ketika objek penghalang berada pada

area 2 maka bunyi *buzzer* lemah dan ketika objek penghalang berada pada area 1 dan 2 maka bunyi *buzzer* keras.

4.2.2 Kebutuhan Non Fungsional

Kebutuhan non fungsional dari sisstem ini terdiri dari kebutuhan perangkat keras dan kebutuhan perangkat lunak yang dijelaskan secara rinci dibawah ini :

4.2.2.1 Kebutuhan Perangkat Keras

Guna mendukung implementasi pembuatan sistem dari sisi perangkat keras maka dilakukan beberapa alat yang dijelaskan berikut ini :

1. Raspberry Pi 3

   Raspberry Pi 3 Sebagai otak pengolahan informasi dalam perancangan sistem
2. Kamera *webcam* logitech C310

   Kamera sebagai sensor yang dibutuhkan untuk mengambil citra video dengan ukuran 320 x 220 piksel
3. Buzzer

   Buzzer digunakan sebagai hasil *output* dari sistem sebagai penanda ada atau tidaknya objek penghalang bagi penyandang tunanetra

4.2.2.2 Kebutuhan Perangkat Lunak

Pada bagian ini terdapat kebutuhan perangkat lunak yang dibutuhkan oleh sistem yaitu opencv untuk mengoperasikan raspberry pi dan untuk menjalankan program lainnya yang dibutuhkan dalam pembuatan program dalam penelitian ini.

1. Windows 7 *Operation System*

   Sitem operasi yang digunakan pada penelitian ini menggunakan Windows 7
2. *Library* OpenCV

   Library OpenCV yang digunakan untuk proses pengolahan citra
3. VNC *Viewer*

   VNC *Viewer* digunakan untuk mengontrol Raspberry Pi 3 melalui latop

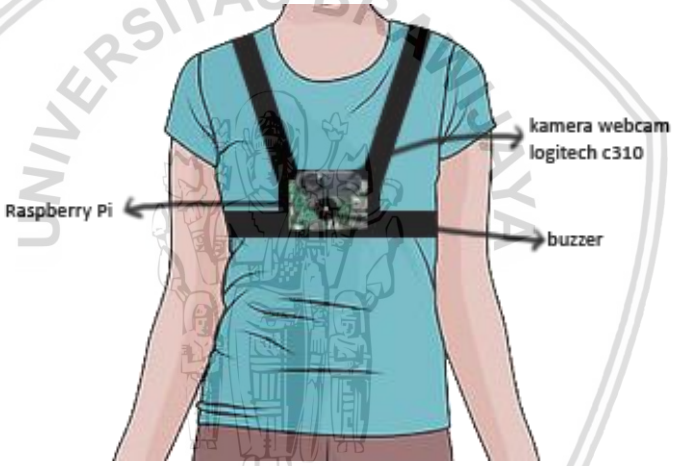
BAB 5 PERANCANGA DAN IMPLEMENTASI

5.1 Perancangan Sistem

Sub-bab ini menjabarkan cara perancangan sistem mulai dari perancangan alat, perancangan perangkat keras hingga perancangan perangkat lunak.

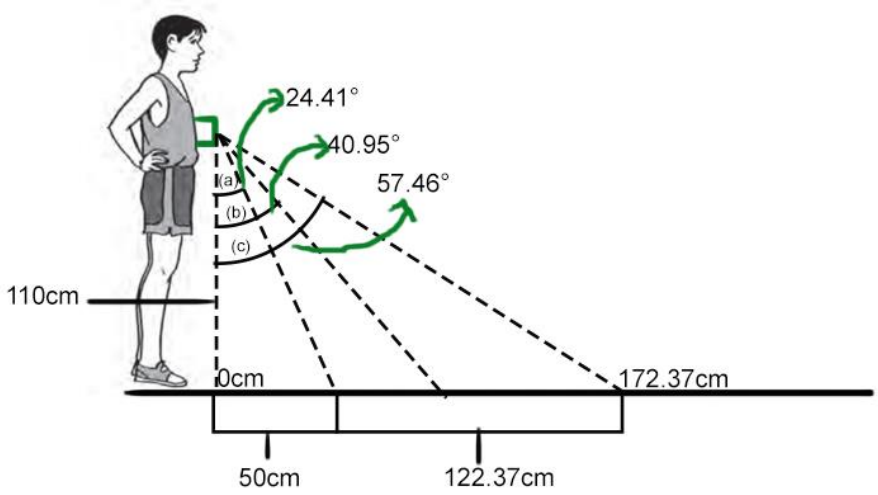
5.1.1 Perancangan Alat Deteksi Objek Penghalang

Dalam perancangan alat untuk mendeteksi objek penghalang perlu diperhatikan peletakan kamera dan sudut kamera agar dapat mengambil citra dengan maksimal. Untuk menggambarkan bentuk *protorype* desain alat ini menggunakan aplikasi Photoshop. Agar perangkat dapat digunakan di ketinggian 110 cm maka digunakan *strap belt chest* yang telah terpasang Raspberry Pi, kamera dan buzzer. *Strap belt chest* ini akan digunakan pada dada depan dan dapat disesuaikan dengan ukuran tubuh pengguna. Perancangan alat dan peletakan sensor dapat dilihat pada Gambar 5.1



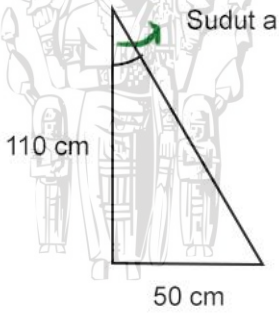
Gambar 5.1 Desain Alat Pendeteksi Objek Penghalang

Perangkat keras yang digunakan pada sistem ini dibagi menjadi tiga bagian, yaitu *input*, pemroses dan *output*. *Input* pada sistem ini menggunakan kamera webcam logitech C310 dan output dari sistem ini yaitu suara dari buzzer. Raspberry Pi sebagai pemroses. Ketiga komponen tersebut dipasangkan berdekatan dengan menggunakan *strip belt chest.* Webcam sebagai sensor kamera dipasang pada bagian dada depan dengan tinggi 110 cm dan sudut kemiringan 41° sehingga cakupan gambar yang dapat diambil yaitu seperti penjelasan dibawah ini:



Gambar 5.2 Perancangan Desain Alat

Pada Gambar 5.2 merupakan desain alat deteksi objek penghalang saat dipasangkan pada dada depan pengguna. Kamera yang terpasang bersama Raspberry Pi dan buzzer digunakan pada ketinggian 110 cm dan sudut kemiringan kamera yaitu 40.95°. Jarak terdekat yang dapat ditangkap oleh kamera yaitu 50 cm di depan pengguna dan jarak terjauh yang dapat ditangkap oleh kamera yaitu 172.37 cm dari depan pengguna



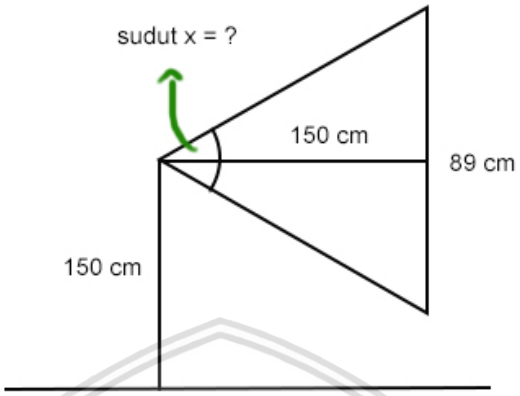
Gambar 5.3 Ilustrasi Sudut a

Adapun penjelasan pada sudut (a) pada Gambar 5.3 yaitu sebesar 24.41° merupakan sudut yang digunakan untuk mencari jarak terdekat yang dapat disorot oleh kamera. Sudut tersebut didapatkan dari perhitungan persamaan 5.1 :

$$\begin{aligned} \tan a &= \frac{50}{110} \\ &= 0{,}454 \\ a &= \arctan 0{,}454 \\ a &= 24{,}41° \end{aligned} \tag{5.1}$$

Sudut 24.41° digunakan untuk men-set jarak terdekat yang dapat dicakup kamera yaitu 50 cm. Sudut tersebut digunakan selanjutnya untuk menentukan sudut kemiringan kamera setelah mengetahui *field of view* dari kamera webcam logitech C310.

Untuk mengetahui *field of view* kamera webcam logitech C310 ini dapat diketahui dengan melakukan perhitungan secara manual menggunakan meteran. Cara ini dilakukan dengan menempatkan kamera pada ketinggian 150 cm dan jarak antara kamera dan dinding yaitu 150 cm.



Gambar 5.4 Ilustrasi *Field of View* Kamera Logitech C310

Pada Gambar 5.4 dapat dilihat bahwa kamera akan menyorot lurus ke depan dinding, kemudian dilakukan pengukuran secara manual titik terendah dan titik tertinggi yang dapat dicakup oleh kamera. Didapatkan bahwa kamera logitech C310 ini dapat menyorot selebar 89 cm. Kemudian melakukan perhitungan untuk mencari sudut *field of view* menggunakan perhitungan pada persamaan 5.2 sebagai berikut :

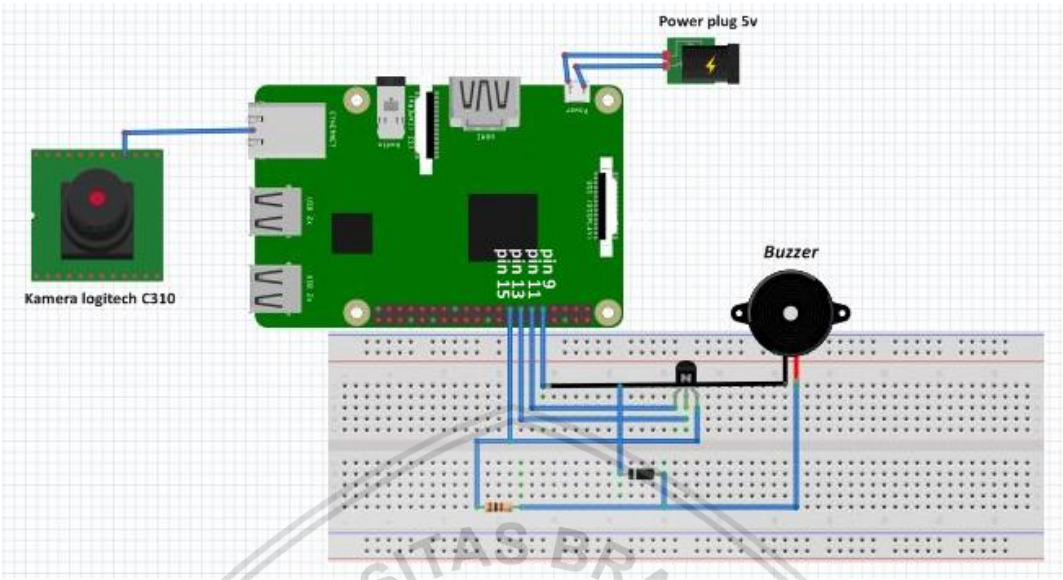
$$\begin{aligned} \tan x_1 &= \frac{\frac{89}{2}}{150} \\ &= \frac{44{,}5}{150} \\ &= 0{,}297 \\ \arctan x_1 &= 0{,}297 \\ &= 16{,}54° \\ x &= 33{,}08° \end{aligned} \tag{5.2}$$

Setelah mendapatkan sudut *field of view* kamera webcam logitech C310 yaitu sebesar 33,08°. Sudut yang diambil yaitu setengah dari *field of view* pada kamera logitech C310 sebesar 16,54°. Sudut tersebut dijumlahkan dengan sudut a (24,41°) menjadi 40,95°. Sudut itulah yang digunakan untuk sudut kemiringan kamera webcam logitech C310 agar dapat menyorot lantai dengan jarak terpendek dari pengguna yaitu 50 cm dan jarak terjauh yang dapat disorot oleh kamera yaitu 173,37 cm

5.1.2 Perancangan Perangkat Keras

Pada tahap perancangan perangkat keras ini menjabarkan skematik rangkaian antara pin pada masing-masing komponen perangkat keras, dinataranya yaitu

Raspberry Pi sebagai pusat kontroler, sensor kamera dan buzzer. Desain rangkaian perancangan alat ini dapat dilihat pada Gambar 5.5.



Gambar 5.5 Diagram Skematik Rangkaian Sistem

Pada Gambar 5.5 dapat dilihat bahwa kamera logitech c310 yang digunakan sebagai sensor untuk mengambil citra gambar dipasangkan pada Raspberry Pi menggunakan kabel USB. Terdapat *buzzer* aktif sebagai indikator peringatan berupa alarm ketika sistem mendeteksi objek penghalang. *Buzzer* dipasangkan pada raspberry pi pada pin GPIO. Serta memerlukan daya yang didapatkan dari powerbank yang dipangkan pada USB OTG. Berikut koneksi antar komponen perangkat yang dapat dilihat lebih jelasnya pada tabel 5.1 dan 5.2

Tabel 5.1 Konfigurasi Pin Kamera dengan Raspberry Pi

Kamera Logitech C310	Raspberry Pi
USB	Slot USB

Dari Tabel 5.1 dapat dilihat bahwa kamera logitech C310 dihubungkan pada Raspberry Pi menggunakan kabel USB pada slot USB Raspberry Pi.

Tabel 5.2 Konfigurasi Pin *Buzzer* dengan Raspberry Pi

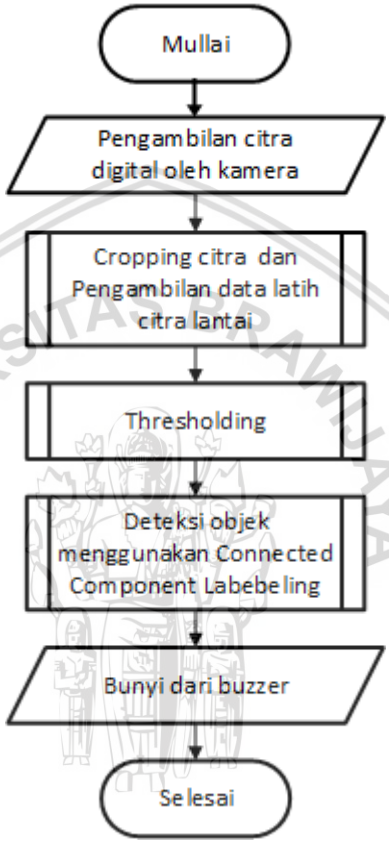
Buzzer	Raspberry Pi
+	Pin 11, 13, 15
-	GND Pin 9

Dari Tabel 5.2 dapat dilihat bahwa untuk dapat mengaktifkan buzzer dipasangkan pada mikrokontroler Raspberry Pi melalui pin GPIO. *Input* + pada *buzzer* dipasangkan pada pin 11, 13 dan 15 pada Raspberry Pi, sedangkan *Input* - pada *buzzer* dipasangkan pada pin 9 pada Raspberry Pi

5.1.3 Perancangan Perangkat Lunak

Pada sub bab perancangan perangkat lunak ini dibagi menjadi beberapa pembahasan mengenai proses pada perangkat lunak yang digunakan yaitu, pengambilan data citra digital, pengambilan data latih citra lantai, perancangan batas nilai threshold, serta perancangan raspberry pi hingga proses deteksi objek penghalang

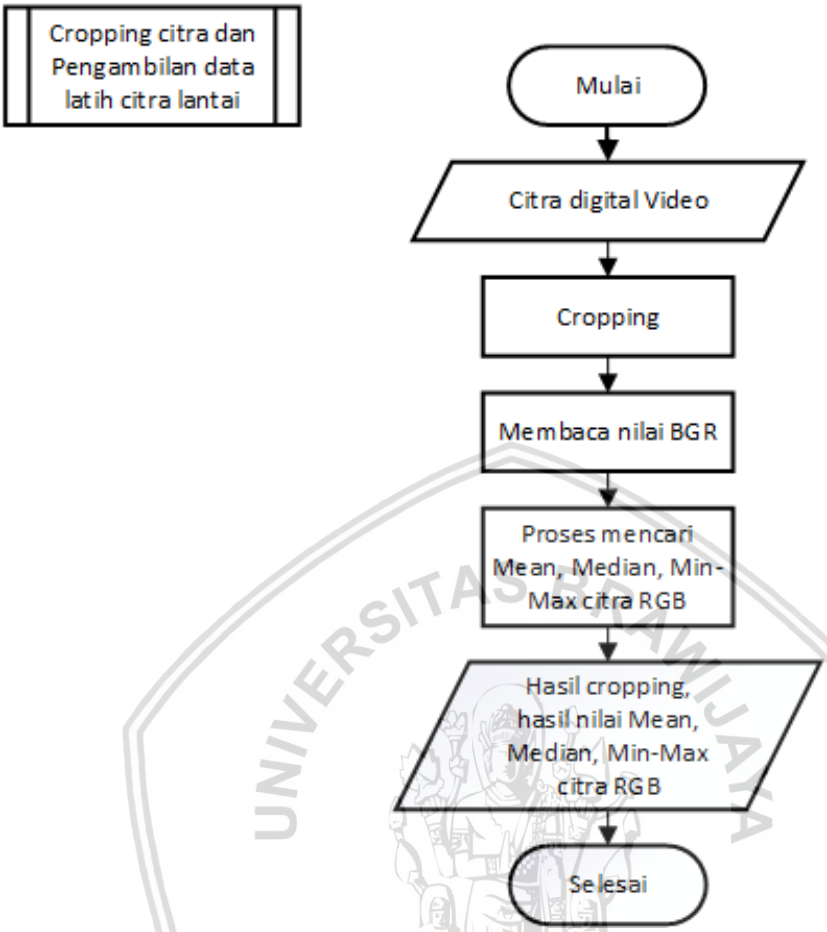
5.1.3.1 Perancangan Proses Utama



Gambar 5.6 Diagram Alir Perancangan Proses Utama

Proses utama dalam sistem ini dapat dilihat pada Gambar 5.6 diatas. Dalam perancangan proses utama tersebut dilakukan beberapa proses yatu, proses pengambilan citra digital oleh kamera dan inisialisasi *variabel* dan *library* yang dibutuhkan terlebih dahulu. Kemudian citra digital tersebut dilakukan pemotongan *(cropping)* pada bagian bawah citra video dengan ukuran tertentu guna mempercepat waktu komputasi. Selanjutnya dilakukan perhitungan nilai *mean, median,* dan *min-max* pada citra BGR pada citra yang sudah dipotong. Setelah itu menentukan nilai batas *threshold* dari hasil perhitungan citra BGR untuk dilakukan proses selanjutnya pemisahan *background* dan *foreground* melalui proses *thresholding.* Setelah mendapatkan hasil dari proses *thresholding,* selanjutnya akan dilakukan proses deteksi objek menggunakan *connected component labeling.* Sistem akan memberikan keluaran berupa bunyi dari *buzzer* ketika sistem dapat berhasil mendeteksi objek penghalang.

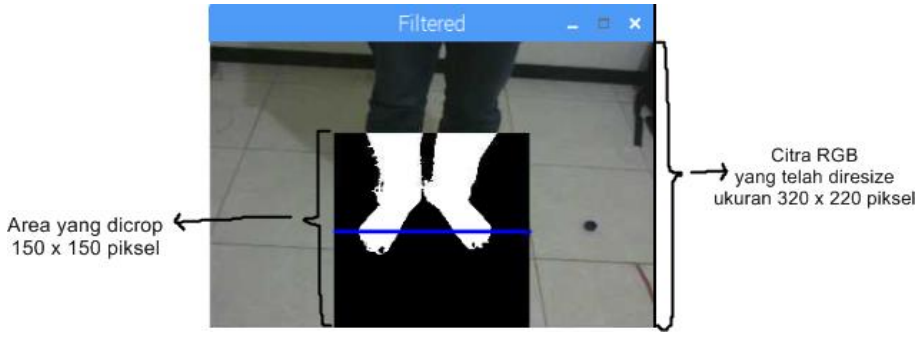
5.1.3.2 Diagram Alir Proses *Cropping* Citra dan Pengambilan Data Latih Lantai



Gambar 5.7 Diagram Alir Proses *Cropping* dan Pengambilan Data Latih Lantai

Diagram Alir pada Gambar 5.7 yaitu menunjukkan alur pada proses *cropping* dan proses pencarian pengambilan data latih lantai. Proses awal dari diagram alir ini yaitu kamera mendapatkan citra dari video yang diolah oleh raspberry. Video tersebut akan di re-*size* pada ukuran 320 x 220 piksel. Kemudian di*crop* pada bagian bawah gambar dengan ukuran 150 x 150 piksel.

Dari hasil *cropping* tersebut diambil data citra masing-masing piksel, yang dimana 1 piksel memiliki 3 parameter warna yaitu *blue, green* dan *red.* Proses pencarian nilai *mean* citra RGB yaitu didapatkan dari nilai rata-rata piksel yang didapatkan dari 1 *frame* gambar. Pencarian nilai *median* citra RGB didapatkan dari nilai tengah dari piksel yang didapatkan dari 1 *frame* gambar. Sedangkan untuk nilai *min-max* citra RGB didapatkan dari nilai terkecil dan terbesar pada piksel-piksel yang terdapat dalam 1 *frame* gambar. Citra video yang telah diproses tersebut dipasangkan kembali dengan video yang asli sebelum dicrop. Sehingga dalam satu frame terlihat seperti Gambar 5.8

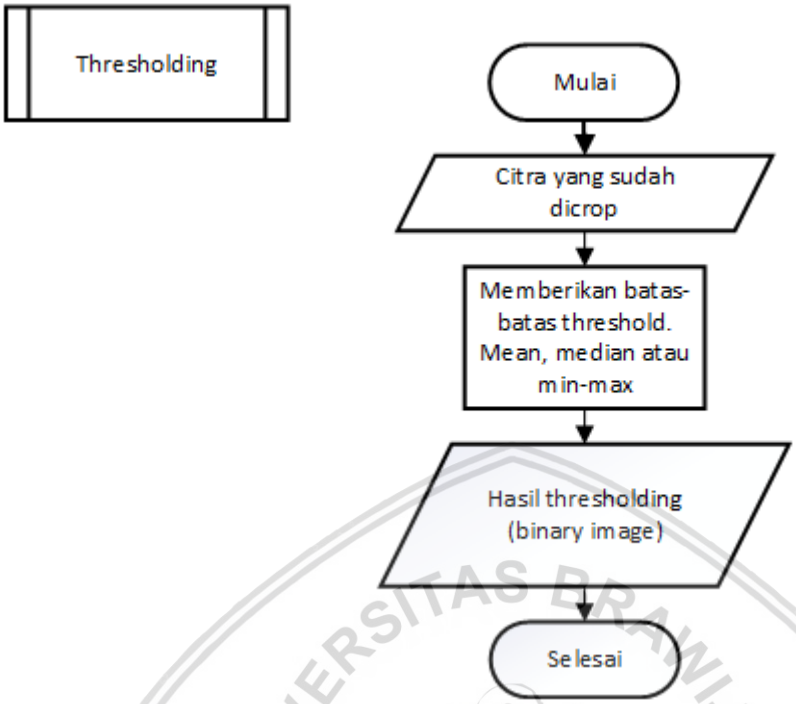


Gambar 5.8 Contoh Frame yang telah di*crop*

Tabel 5.3 Data Latih Lantai

WAKTU	*Red Max*	*Red Min*	*Green Max*	*Green Min*	*Blue Max*	*Blue Min*
Pagi 1	198	74	204	89	228	74
Pagi 2	221	68	205	74	201	56
Pagi 3	179	81	184	82	214	57
Pagi 4	194	88	212	91	233	71
Pagi 5	235	60	223	74	229	47
Pagi 6	192	77	195	78	193	68
Pagi 7	192	77	195	78	193	68
Pagi 8	176	77	177	82	187	81
Pagi 9	181	82	185	87	189	81
Pagi 10	174	80	189	85	184	79
Siang 1	164	88	170	93	174	87
Siang 2	168	87	172	92	178	86
Siang 3	167	96	172	98	171	93
Siang 4	188	92	190	97	205	90
Siang 5	236	74	254	93	255	81
Siang 6	142	66	144	71	139	65
Siang 7	142	69	142	73	137	66
Siang 8	198	60	214	61	199	53
Siang 9	152	48	152	49	147	40
Siang 10	180	77	181	76	178	71
Malam 1	222	72	225	72	230	71
Malam 2	252	90	255	95	255	93
Malam 3	241	84	249	89	253	88
Malam 4	214	70	220	75	254	72
Malam 5	186	66	190	72	195	63
Malam 6	183	70	194	75	181	71
Malam 7	198	77	212	82	218	78
Malam 8	152	71	157	76	158	71
Malam 9	241	71	253	76	255	70
Malam 10	235	66	244	71	246	68

5.1.3.3 Diagram Alir Proses *Thresholding*



Gambar 5.9 Diagram Alir Proses *Thresholding*

Proses perancangan perangkat lunak proses *thresholding* ditunjukkan pada gambar 5.9 yaitu dengan masukan citra RGB yang sudah di*crop.* Data latih citra yang telah didapatkan kemudian dihitung dan dicari nilai *mean, median* dan *min-maxnya*. Contoh perhitungan mencari nilai batas *threshold mean, median* dan *min-max* yaitu sebagai berikut :

a. Contoh manualisasi perhitungan *threshold mean*

Tabel 5.4 Contoh perhitungan parameter *threshold mean*

No.	Data Citra Lantai (RGB)					
	Red Max	*Red Min*	*Green Max*	*Green Min*	*Blue Max*	*Blue Min*
1.	198	74	204	89	228	74
2.	221	68	205	74	201	56
3.	179	81	184	82	214	57
4.	194	88	212	91	233	71
5.	235	60	223	74	229	47
6.	193	68	195	78	192	77
Mean	203,3333	73,16667	203,8333	81,33333	216,1667	63,66667

$$mean = \frac{1}{n}\sum_{i=1}^{n} x_1$$

Keterangan :
n : Jumlah data
x : nilai data ke i

$$Mean\ \ Red\ Max = \frac{198 + 221 + 179 + 194 + 235 + 193}{6}$$
$$= \frac{1220}{6}$$
$$= 203{,}33$$

b. Contoh manualisasi perhitungan *threshold median*

Tabel 5.5 Contoh Perhitungan Parameter *threshold Median*

No.	Data Citra Lantai (RGB)					
	Red Max	*Red Min*	*Green Max*	*Green Min*	*Blue Max*	*Blue Min*
1.	198	74	204	89	228	74
2.	221	68	205	74	201	56
3.	179	81	184	82	214	57
4.	194	88	212	91	233	71
5.	235	60	223	74	229	47
6.	193	68	195	78	192	77
Median	196	71	204,5	80	221	64

$$Median = \frac{1}{2}\left(x_{\left(\frac{n}{2}\right)} + x_{\left(\frac{n}{2}+1\right)}\right)$$

Keterangan
n : Jumlah data
x : nilai data
Data nilai Red Max = 198, 221, 179, 194, 235, 193
Data nilai Red Max setelah diurutkan = 179. 193, 194, 198, 221, 235

$$Median\ Red\ Max = \frac{1}{2}\left(x_{\left(\frac{6}{2}\right)} + x_{\left(\frac{6}{2}+1\right)}\right)$$
$$= \frac{1}{2}\left(x_3 + x_4\right)$$
$$= \frac{1}{2}\left(194 + 198\right)$$
$$= \frac{1}{2}\left(392\right)$$
$$= 196$$

c. Contoh manualisasi perhitungan *threshold min-max*

Tabel 5.6 Contoh Perhitungan Parameter *Threshold Min-max*

No.	Data Citra Lantai (RGB)					
	Red Max	*Red Min*	*Green Max*	*Green Min*	*Blue Max*	*Blue Min*
1.	198	74	204	89	228	74
2.	221	68	205	74	201	56
3.	179	81	184	82	214	57
4.	194	88	212	91	233	71
5.	235	60	223	74	229	47
6.	193	68	195	78	192	77
Min-Max	235	60	223	74	233	47

Berdasarkan contoh manualisasi perhitungan parameter *threshold mean, median* dan *min-max* yang telah dihitung pada data latih lantai didapatkan batas-batas thershold sebagai berikut :

Tabel 5.7 Batas-batas Parameter *Threshold*

Data	Data Citra Lantai (RGB)					
	Red Max	*Red Min*	*Green Max*	*Green Min*	*Blue Max*	*Blue Min*
MEAN	193,4	75,2	198,6	80,2	202,6	71,9
MEDIAN	190	75,5	194,5	78	197	71
MaxMin	252	48	255	49	255	40

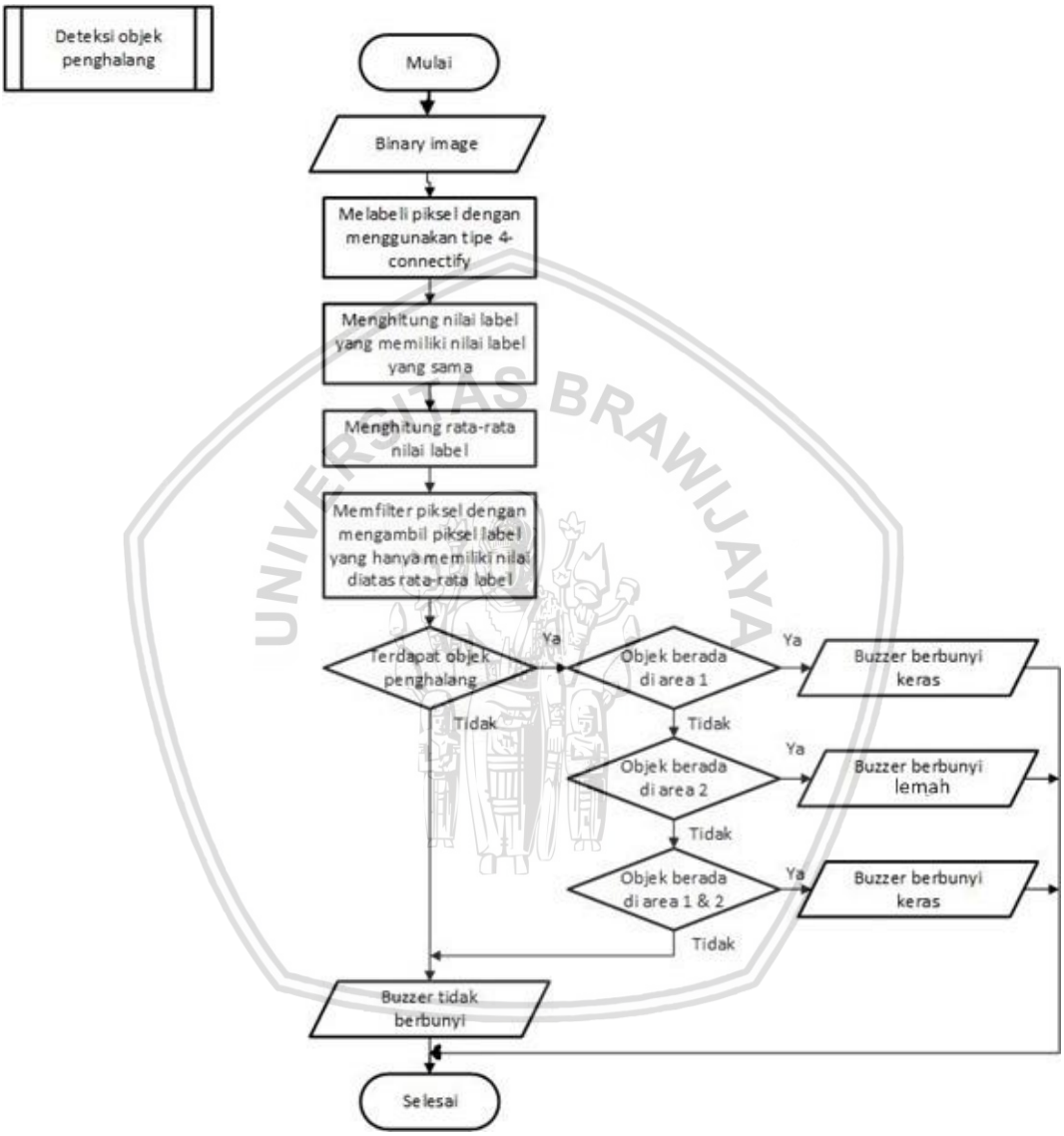
Pada Tabel 5.7 dapat dilihat bahwa batas parameter *threshold mean* yang digunakan yaitu citra RGB dengan nilai minimal piksel *red* 75.2, *green* 80.2, *blue* 71.9 dan maksimal piksel *red* 193.4, *green* 198.6 dan *blue* 202.6. Batas parameter *threshold median* yang digunakan yaitu citra RGB dengan nilai minimal piksel *red* 75.5, *green* 78, *blue* 71 dan maksimal piksel *red* 190, *green* 194.5 dan *blue* 197. Sedangkan parameter *threshold min-max* yang digunakan yaitu citra RGB dengan nilai minimal piksel *red* 48, *green* 49, *blue* 40 dan maksimal piksel *red* 252, *green* 255 dan *blue* 255.

5.1.3.4 Perancangan Deteksi Objek Penghalang

Untuk melakukan proses deteksi objek penghalang menggunakan *Connected Component Labeling* sesuai pada Gambar 5.10 dimana yang menjadi masukan awal yaitu *binary image* dari proses *thresholding*. Sebuah objek merupakan kumpulan dari piksel-piksel yang saling berdekatan, maka metode metode *connected component labeling* pada penelitian ini yaitu untuk menandai piksel tetangga yang memiliki komponen piksel yang sama dengan tipe 4-*connectivity* yaitu mengecek 4 piksel tetangga di atas, bawah dan samping kiri kanan.

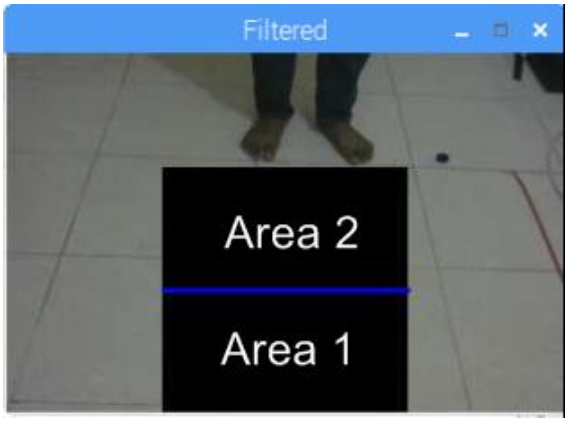
Pengecekan piksel ketetetanggaan ini sekaligus menghitung jumlah pixel label yang memiliki label yang sama. Dari nilai-nilai tersebut didapatkan nilai rata-rata piksel. Nilai rata-rata piksel tersebut digunakan sebagai batas filtering objek.

Sistem dapat mendeteksi objek penghalang jika objek pada sistem memiliki nilai piksel senilai atau diatas rata-rata piksel label. Obek yang memiliki nilai piksel label kurang dari rata-rata akan diabaikan dan dianggap sebagai bukan objek. Ketika sistem mendeteksi objek dengan nilai piksel senilai dan atau diatas nilai rata-rata piksel maka *buzzer* akan berbunyi. Bunyi *buzzer* dibagi menjadi 2 yaitu bunyi *buzzer* keras dan bunyi *buzzer* lemah.



Gambar 5.10 Diagram alir perancangan deteksi objek penghalang

Pada Gambar 5.10 merupakan tampilan sistem yang menunjukkan area 1 dan area 2. Ketika objek terlihat pada area 1 maka dapat direpresentasikan bahwa objek telah dekat dengan pengguna sehingga *buzzer* akan berbunyi keras. Ketika objek terlihat pada area 2 saja maka direpresentasikan bahwa objek masih tidak begitu dekat dengan pengguna maka *buzzer* akan berbunyi lemah. Namun ketika objek terlihat pada area 1 dan 2 maka *buzzer* akan berbunyi keras.



Gambar 5.11 Area 1 dan Area 2 pada Sistem

5.2 Implementasi Sistem

Implementasi sistem merupakan tahap untuk merealisasikan pembuatan sistem berdasarkan semua perancangan yang telah dilakukan sebelumnya. Pada sub bab ini menjelaskan satu per satu secara rinci terkait implementasi alat, implementasi perangkat keras serta implementtasi perangkat lunak.

5.2.1 Implementasi Alat Deteksi Objek Penghalang

Dalam mengimplementasikan alat deteksi objek penghalang ini mengacu pada subbab 5.1.1 yaitu desain alat deteksi objek penghalang. Alat yang digunakan yaitu *strip belt chest* yang dipasangkan pada bagian dada depan dengan tinggi 110 cm. Hasil Implementasi alat beserta peletakan mikrokontroler dan sensor yang digunakan ditunjukkan pada Gambar 5.12

(a) (b)

Gambar 5.12 Implementasi Alat Deteksi Objek Penghalang

(a) Implementasi alat deteksi objek penghalang tampak depan;
(b) implementasi alat deteksi objek penghalang tampak belakang.

5.2.2 Implementasi Perangkat Keras

Tahap ini imenjelaskan proses pengimplementasian perangkat keras yang mencangkup komponen elektronik diantara lain Raspberry Pi, sensor kamera Logitech C310 dan *buzzer*. Keseluruhan komponen dirangkai menjadi satu yang dihubungkan berdasarkan perancangan yang telah dijelaskan pada subbab 5.1.2

Gambar 5.13 Implementasi Perangkat Keras

Pada Gambar 5.13 menunjukkan hasil implementasi sensor kamera, *buzzer* dan Raspberry Pi yang telah dirangkai dan dipasangkan pada *strip belt chest*.

5.2.3 Implementasi Perangkat Lunak

Pada implemantasi perangkat lunak akan dijelaskan mengenai realisasi dari perancangan perangat lunak yang telah dibahas pada sub bab perancangan perangkat lunak. Pada implementasi perangkat lunak ini berupa kode program pada controller menggunakan aplikasi openCV dengan bahasa python 3.5

5.2.3.1 Implementasi Proses Utama

Implementasi proses utama yaitu membahas tahap proses utama yang dilakukan dalam beberapa proses yang berhubungan dengan pengambilan citra digital oleh kamera, inisialisasi *variabel* dan *library* yang dibutuhkan, melakukan pemotongan citra (*cropping)*, melakukan perhitungan nilai *mean, median,* dan *min-max* pada citra BGR, menentukan nilai batas *threshold* dan proses deteksi objek menggunakan *connected component labeling.* Kode program yang diimplementasikan berhubungan dengan proses-proses utama dapat dilihat pada tabel 5.8

Tabel 5.8 Kode Program Inisialisasi *Library* dan *variable*

Baris	Kode Program
1.	`import cv2`
2.	`import numpy as np`
3.	`import time`
4.	`import collections`
5.	`import RPi.GPIO as GPIO`
6.	`GPIO.setmode(GPIO.BOARD)`
7.	`GPIO.setup(11,GPIO.OUT)`
8.	`GPIO.output(11,GPIO.LOW)`
9.	`GPIO.setup(13,GPIO.OUT)`

```
10.  GPIO.output(13,GPIO.LOW)
11.  GPIO.setup(15,GPIO.OUT)
12.  GPIO.output(15,GPIO.LOW)
```

Pada implementasi proses utama membutuhkan beberapa *library* OpenCV untuk menunjang proses sistem agar dapat berjalan dengan maksimal. Pada awal kode program, diinisialisasikan terlebih dahulu beberapa *library* OpenCV yang digunakan. Pada baris ke-1 "cv2" yaitu digunakan untuk memanggil *library* cv2 atau *library* OpenCv. Baris ke-2 "*numpy as np"* yaitu *library* yang digunakan untuk mengolah angka pada OpenCV. Baris ke-3 *library* "*time"* untuk timer delay. Baris ke-4 *library* "*collections"* digunakan untuk alat menghitung dan mendukung perhitungan agar lebih mudah dan cepat dan baris ke-5 "Rpi.GPIO" *library* untuk mengontrol pin-pin raspberry pi.

Selain menginisialisasi *library* yang dibutuhkan juga menginisialisasi *variable* diataranya yaitu pada baris ke-6 "GPIO.setmode(GPIO.BOARD)" digunakan untuk skema penomoran dari raspberry pi. Baris ke-7, 9 dan 11 "GPIO.setup(11,GPIO.OUT)", "GPIO.setup(13,GPIO.OUT)" dan "GPIO.setup(15,GPIO.OUT)" yaitu pin 11, 13 dan 15 pada raspberry pi sebagai pin output set buzzer berbunyi sedangkan pada baris ke-8, 10 dan 12 "GPIO.output(11,GPIO.LOW)", "GPIO.output(13,GPIO.LOW)", dan variable "GPIO.output(15,GPIO.LOW)" yaitu pin 11, 13 dan 15 pada raspberry sebagai output buzzer untuk menset buzzer tidak berbunyi.

Setelah melakukan inisialisasi *library* dan *variable*, langkah selanjutnya yaitu mengambil citra video dari kamera. Untuk membaca citra dari kamera dapat dilihat pada kode program pada Tabel 5.9 yaitu pada baris ke-1 hingga baris ke-5 yaitu perintah yang digunakan untuk mengambil citra video.

Tabel 5.9 Kode Program Pembacaan Citra Video dari Kamera

Baris	Kode Program
1.	`cap = cv2.VideoCapture(0);`
2.	`while(cap.isOpened()):`
3.	`    ret, frame = cap.read()`
4.	`    if ret is False:`
5.	`        break`

5.2.3.2 Implementasi *Cropping* Citra dan Pengambilan Data Latih Lantai

Pada implementasi *cropping* citra dan pengambilan data latih lantai ini dapat dilihat pada kode program Tabel 5.10 :

Tabel 5.10 Kode Program *Cropping* Citra

Baris	Kode Program
1.	`frame=cv2.resize(frame,(340,220))`
2.	`    t = frame.shape[0]`
3.	`    p = frame.shape[1]`
4.	`    rgb = frame[(t - 150): t, (int(p/2) - 75):(int(p/2) + 75)]`

Baris pertama pada Tabel 5.10 yaitu me*resize* ukuran citra video menjadi lebih kecil 320 x 220 piksel guna mempercepat waktu komputasi. Baris ke-2-4 yaitu memotong bagian bawah citra video dengan ukuran 150 x 150 piksel untuk dijadikan piksel yang akan digunakan sebagai masukan pada proses selanjutnya.

Proses selanjutnya yaitu mencari data latih lantai. Data latih ini didapatkan dari mengambil 30 sampel gambar lantai masing-masing 10 video pada waktu pagi, siang dan malam. Dari gambar lantai tersebut diambil nilai maksimal dan minimal untuk setiap parameter warna *red, green* dan *blue* pada 1 kali *frame.* Maka didapatkan nilai rMax, rMin, bMax, bMin, gMax dan gMin. Kode program untuk mencari nilai data latih lantai dapat dilihat pada tabel 5.11 :

Tabel 5.11 Kode program Mencari Data Latih Lantai

Baris	Kode Program
1.	`bMax = 0`
2.	`bMin = 255`
3.	`gMax = 0`
4.	`gMin = 255`
5.	`rMax = 0`
6.	`rMin = 255`
7.	`rgb = frame[(t - 150): t, (int(p/2) - 75):(int(p/2) + 75)]`
8.	`for x in rgb:`
9.	`for y in x:`
10.	`blue = y[0]`
11.	`green = y[1]`
12.	`red = y[2]`
13.	`if bMax < blue:`
14.	`bMax = blue`
15.	`if bMin > blue:`
16.	`bMin = blue`
17.	`if gMax < green:`
18.	`gMax = green`
19.	`if gMin > green:`
20.	`gMin = green`
21.	`if rMax < red:`
22.	`rMax = red`
23.	`if rMin > red:`
24.	`rMin = red`
25.	`print ("bMax = " + str(bMax))`
26.	`print ("gMax = " + str(gMax))`
27.	`print ("rMax = " + str(rMax))`
28.	`print ("bMin = " + str(bMin))`
29.	`print ("gMin = " + str(gMin))`
30.	`print ("rMin = " + str(rMin))`

Pada baris ke-1 hingga baris ke-6 yaitu menginisialisasikan *variable* bMax, gMax dan rMax untuk set awal nilai *blue max, green max* dan *red max*nya yaitu 0. *Variable* bMin, gMin dan rMin untuk set awal nilai *blue min, green min* dan *red min*nya yaitu 255. Baris ke-7 yaitu ptoses cropping video menjadi ukuran 150x150 piksel. Piksel dari hasil cropping tersebutlah yang dicari nilai maksimal dan minimal citra RGB. Baris 8-12 yaitu membuat array untuk diisikan nilai citra RGB pada masing-masing array *blue, green*, dan *red.* Baris ke-13 hingga 24 merupakan proses pencarian nilai maksimal dan minimal masing-masing nilai warna *blue, green* dan

red dari data gambar lanti. Baris 25-30 digunakan untuk menampilkan nilai maksimal dan minimal nilai warna *blue, green* dan *red.*

5.2.3.3 Implementasi Proses *Thresholding*

Implementasi Proses *thresholding* ini yaitu merubah citra RGB menjadi *grayscale* atau *binary image*. Setelah itu diberikan batas-batas nilai *threshold* untuk memisahkan *background* dengan *foregroun*d. Nilai *threshold* didapatkan dengan menghitung nilai *mean, median,* dan *min-max* dari setiap komponen citra RGB. Kode program *thresholding* dapat dilihat pada Tabel 5.12 :

Tabel 5.12 Kode Program *Thresholding*

Baris	Kode Program
1.	`#threshold MEAN`
2.	`Maximum = np.array([202.6,198.2,193.4])`
3.	`Minimum = np.array([71.9,80.2,75.2])`
4.	`#threshold MEDIAN`
5.	`Maximum = np.array([197.0,194.5,190.0])`
6.	`Minimum = np.array([71.0,78.0,75.0])`
7.	`#threshold MinMax`
8.	`#Maximum = np.array([255,255,252])`
9.	`#Minimum = np.array([40,49,48])`
10.	`Tampil = cv2.inRange(rgb, Minimum, Maximum)`
11.	`Hasil = cv2.bitwise_not(Tampil)`

Baris pertama hingga ke-3 yaitu batas nilai parameter *threshold* untuk *mean,* baris ke-4 hingga ke-6 yaitu batas nilai *threshold* untuk *median* dan baris ke-7 hingga ke-0 merupakan batas threshold untuk nilai min-max. Sedangkan pada baris ke-10 yaitu memasukkan rumus batas *threshold* dan baris ke-11 membalik nilai hasil *threshold*. Jika pada awalnya hasil dari *thseshold* yaitu *background* bernilai 1 dan objek bernilai 0, maka pada baris ke-11 merubah nilai *background* menjadi 0 dan objek menjadi 1.

5.2.3.4 Implementasi Deteksi Objek Penghalang

Implementasi deteksi objek penghalang menggunakan *Connected Component Labeling* ini sesuai pada perancangan yang sudah dijelaskan pada sub bab perancangan deteksi objek penghalang. dimana yang menjadi masukan awal yaitu *binary image* dari proses *thresholding*. Disini metode *connected component labeling* yaitu untuk menandai piksel tetangga yang memiliki komponen piksel yang sama dengan tipe 4-*connectivity* yaitu mengecek 4 piksel tetangga di atas, bawah dan samping kiri kanan. Setelah melabeli piksel, selanjutnya menghitung nilai label yang memiliki label yang sama. Dari nilai-nilai tersebut diambil nilai rata-ratanya untuk memfilter objek dan membandingkan nilai piksel objek dengan nilai rata-rata piksel objek yang telah dilabeling. Ketika objek terdeteksi pada area 1 saja maka *buzzer* akan berbunyi keras, ketika objek terdeteksi pada area 2 saja maka *buzzer* akan berbunyi lemah sedangkan ketika objek terdeteksi pada area 1 dan 2 maka *buzzer* akan berbunyi keras. Kode program deteksi objek penghalang menggunakan *connected component labeling* dapat dilihat pada tabel 5.13 berikut ini :

Tabel 5.13 Kode Program Deteksi Objek Penghalang

Baris	Kode Program
1.	`ret, labels = cv2.connectedComponents(Hasil,`
2.	`connectivity=4)`
3.	`    label1d = labels.ravel()`
4.	`    counter = collections.Counter(label1d)`
5.	`    counter_array = np.array(list(counter.items()))[1:]`
6.	`    rata_array = np.mean(counter_array[:,1])`
7.	`    counter_array[counter_array[:,1]<rata_array] = 0`
8.	`    f = counter_array[counter_array[:,0]!=0][:,0]`
9.	`    label = labels.copy() * 0`
10.	`    for x in f:`
11.	`        label[labels==x] = 255`
12.	`    label_hue = np.uint8(179*labels/np.max(labels))`
13.	`    blank_ch = 255*np.ones_like(label_hue)`
14.	`    labeled_img = cv2.merge([label_hue, blank_ch,`
15.	`blank_ch])`
16.	`    labeled_img[label_hue==0] = 0`
17.	`    label_filtered = np.uint8(label)`
18.	`    frame_ccl = frame.copy()`
19.	`    frame_ccl[(t - 150): t, (int(p/2) - 75):(int(p/2) +`
20.	`75)] = label_filtered`
21.	`    cv2.line(frame_ccl, ((int(p/2)-75),(t-75)),`
22.	`((int(p/2)+75),(t-75)), (255,0,0), 2)`
23.	`    end_time = millis() - start_time`
24.	`    cv2.imshow("Filtered",frame_ccl)`

Baris 1-3 merupakan fungsi connected component labeling 4 ketetanggan yang dimana melabeli piksel dengan melihat 4 piksel tetangganya. Baris 3-4 membuat array untuk menyimpan nilai piksel yang telah dilabeli. Baris 5-6 untuk menghitung nilai piksel ketetanggan. Baris ke-7 dan 8 yaitu untuk menentukan membandingkan nilai piksel dengan nilai rata-rata piksel label. Objek dikatakan penghalang jika nilai ketetanggaan pikselnya memiliki nilai setara atau lebih dari nilai rata-rata piksel label dan objek yang memiliki nilai dibawah rata-rata piksel akan diabaikan atau bukan dianggap sebagai objek penghalang. Baris ke 9-24 proses melabeli piksel ketetanggan dengan warna yang berbeda.

Tabel 5.14 Kode program bunyi *buzzer* keras dan lemah

Baris	Kode Program
1.	`slice_top = label_filtered[0:75,0:150]`
2.	`slice_bot = label_filtered[75:150,0:150]`
3.	`    count_top =`
	`collections.Counter(np.ravel(slice_top))[255]`
4.	`    count_bot =`
	`collections.Counter(np.ravel(slice_bot))[255]`
5.	`    if count_top > 0  and count_bot <= 0:`
6.	`        GPIO.output(11, GPIO.HIGH)`
7.	`        GPIO.output(13, GPIO.LOW)`
8.	`        GPIO.output(15, GPIO.HIGH)`
9.	`    elif count_bot > 0:`
10.	`        GPIO.output(11, GPIO.HIGH)`
11.	`        GPIO.output(15, GPIO.LOW)`
12.	`        GPIO.output(13, GPIO.HIGH)`
13.	`    else :`

```
14.             GPIO.output(11, GPIO.LOW)
15.             GPIO.output(13, GPIO.LOW)
16.             GPIO.output(15, GPIO.LOW)
```

Pada tabel 5.14 pada baris ke-1 dan 2 yaitu memberikan garis pembatas antara area 1 dan 2, dimana area 1 yaitu berada di piksel ke-0 hingga 75 dan area 2 berada pada piksel ke-76 hingga ke-150. Baris ke-3 dan 4 untuk menghitung piksel objek berada pada area 1 atau area 2. Baris ke-5 hingga ke 8 yaitu untuk membunyikan *buzzer* keras. Baris ke-9-12 yaitu untuk membunyikan *buzzer* lemah dan baris ke-13 sampai 16 yaitu untuk membunyikan *buzzer* ketika objek berada di area 1 dan 2.

BAB 6 PENGUJIAN DAN ANALISLIS

6.1 Pengujian dan Analisis nilai *Threshold* pada Berbagai Waktu dan Jarak yang Berbeda-beda

6.1.1 Tujuan

Pengujian ini dilakukan untuk mengetahui nilai *threshold* yang memiliki akurasi paling baik dalam mendeteksi objek penghalang. Terdapat tiga nilai *threshold* yaitu nilai *mean, median* dan *min-max.* Pengujian nilai *threshold* ini dilakukan pada 5 objek berbeda yaitu orang, *box*, tempat sampah, meja dan pintu. Pada pengujian ini, objek berada pada jarak tertentu yaitu 50 cm, 75 cm, 100 cm dan 125 cm di depan pengguna.

6.1.2 Prosedur Pengujian

Prosedur pengujian dilakukan agar pengujian dapat berjalan dengan semestinya, adapun langkah-langkah yang dilakukan adalah sebagai berikut :

1. Melakukan pemasangan *strap belt chest* yang sudah terpasang Raspberry pi, kamera dan buzzer pada dada depan pengguna dengan ketinggian 110 cm.
2. Menghubungkan Raspberry dengan sumber daya.
3. Membuka aplikasi VNC untuk membuka program dan menjalankan program.
4. Memberikan input berupa 5 objek berbeda dan pada jarak yang berbeda-beda (50 cm, 75 cm, 100 cm, dan 125 cm).
5. Mengamati dan mengambil data hasil keluaran sistem.
6. Menentukan tingkat keberhasilan sistem dengan mendeteksi rata-rata akurasi jarak tiap waktu.

6.1.3 Hasil dan Analisis Pengujian

6.1.3.1 Threshold dengan Nilai *Mean*

Dari data hasil perhitungan RGB yang telah disajikan pada bab 5, maka nilai *threshold* yang digunakan adalah Bmin = 71.9, Bmax = 202.6, Gmin = 80.2 dan Gmax = 198.2, Rmin = 75.0 dan Rmax = 193.4. Hasil pengujian parameter nilai *threshold* dengan nilai *mean* dapat dilihat pada Tabel 6.1

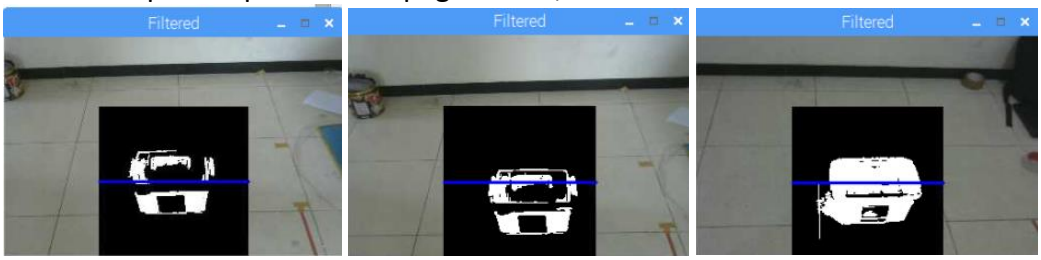
Tabel 6.1 Hasil Pengujian Parameter *Threshold* Nilai M*ean*

No	Objek	Jarak	Jarak Hasil Deteksi (cm)			Akurasi %		
			Pagi	Siang	Malam	Pagi	Siang	Malam
1	*Box*	50 cm	50	50	52	100	100	96
2	Meja	50 cm	50	50	50	100	100	100

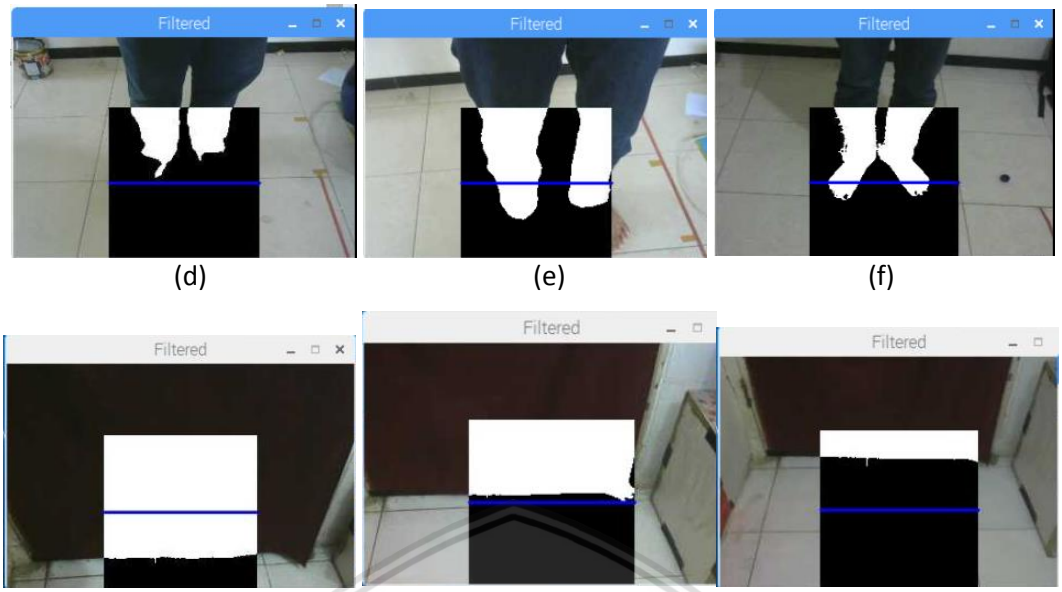
3	Orang	50 cm	70	70	45	71	71	90
4	Dinding	50 cm	50	50	50	100	100	100
5	Pintu	50 cm	0	0	0	0	0	0
Rata-rata jarak tiap waktu (%)						**74,2**	**74,2**	**77,2**
6	*Box*	75 cm	75	75	61	100	100	81.3
7	Meja	75 cm	75	75	75	100	100	100
8	Orang	75 cm	90	75	75	83.3	100	100
9	Dinding	75 cm	63	75	75	84	100	100
10	Pintu	75 cm	90	75	58	83.3	100	77.3
Rata-rata jarak tiap waktu (%)						**90,12**	**100**	**94.32**
11	*Box*	100 cm	97	100	100	97	100	100
12	Meja	100 cm	100	100	100	100	100	100
13	Orang	100 cm	100	100	100	100	100	100
14	Dinding	100 cm	100	100	100	100	100	100
15	Pintu	100 cm	100	100	75	100	100	75
Rata-rata jarak tiap waktu (%)						**99,4**	**100**	**95**
16	*Box*	125 cm	125	124	125	100	99,2	100
17	Meja	125 cm	125	121	125	100	98,8	100
18	Orang	125 cm	125	125	44	100	100	35,2
19	Dinding	125 cm	125	125	125	100	100	100
20	Pintu	125 cm	125	0	100	100	0	80
Rata-rata jarak tiap waktu (%)						**100**	**74,7**	**83,04**
Akurasi tiap waktu %						**90,93**	**87,22**	**87,39**
Rata-rata akurasi seluruh pengujian (%)						**88,51**		

Jarak hasil deteksi didapatkan dari pengukuran secara manual antara jarak objek sesungguhnya dengan jarak objek yang terlihat pada hasil sistem. Citra yang telah di proses dipengaruhi beberapa hal seperti *noise*, warna yang tidak dapat terdeteksi pada batas threshold yang mengakibatkan objek pada jarak sesungguhnya tidak sama dengan jarak yang terlihat pada sistem. Sehingga pada penelitian ini pengujian jarak dilakukan secara manual yaitu mengukur jarak objek sesungguhnya dan mengukur jarak objek setelah diproses menggunakan meteran.

Pada tabel 6.1 dapat dilihat bahwa dengan menggunakan nilai *threshold mean* mendapatkan akurasi sebesar 88,51%. *Threshold* menggunakan parameter ini memiliki nilai akurasi cukup baik pada waktu pagi, siang dan malam. Akurasi terbaik didapatkan pada waktu pagi hari 90,93% .



(a) (b) (c)



(d) (e) (f)

(g) (h) (i)

Gambar 6.1 Hasil pengujian objek *box,* orang dan dinding menggunakan *threshold mean*

(a) Pengujian objek *box* pada pagi hari; (b) Pengujian objek *box* pada siang hari; (c) Pengujian objek *box* pada malam hari; (d) Pengujian objek orang pada pagi hari; (e) Pengujian objek orang pada siang hari; (f) Pengujian objek orang pada malam hari; (g) Pengujian objek dinding pada pagi hari; (h) Pengujian objek dinding pada siang hari; (i) Pengujian objek dinding pada malam hari;

6.1.3.2 *Threshold* dengan Nilai *Median*

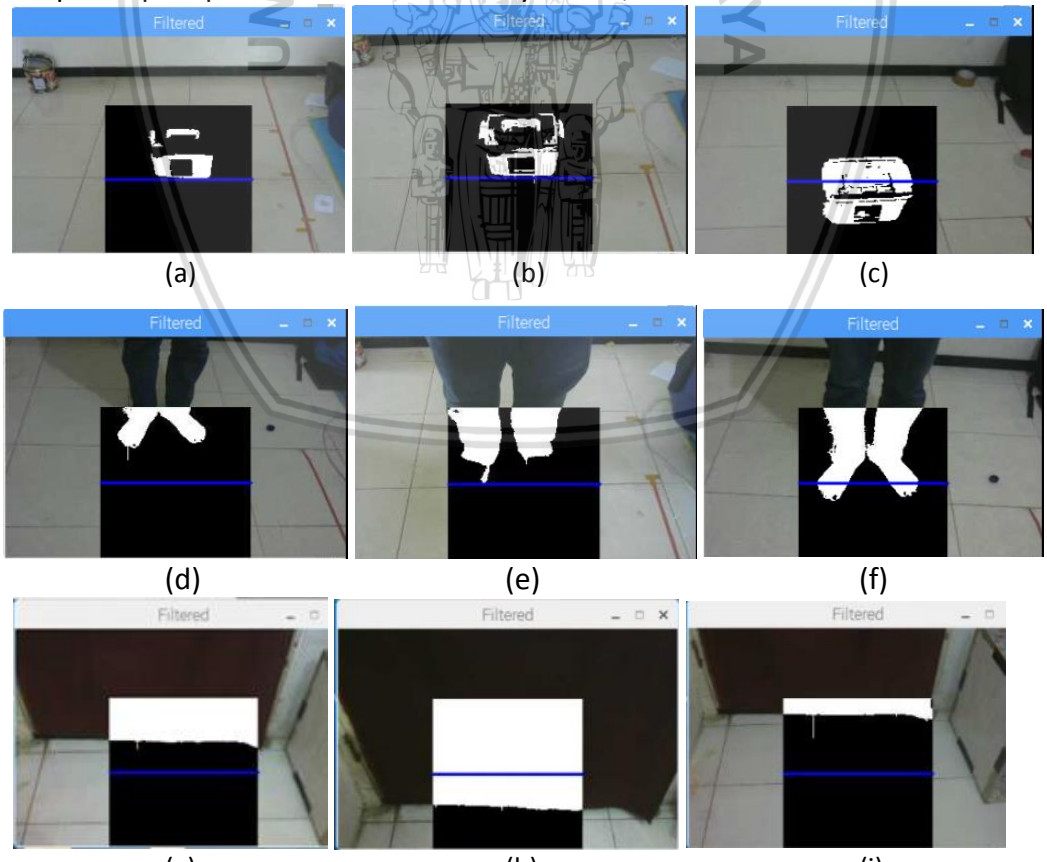
Dari data hasil perhitungan BGR yang telah disajikan pada bab 5, maka nilai *threshold* yang digunakan adalah Bmin = 71.0, Bmax = 179.0, Gmin = 78.0 dan Gmax = 194.5, Rmin = 75.0 dan Rmax = 190.0. Hasil pengujian parameter nilai *threshold* dengan nilai *mean* dapat dilihat pada Tabel 6.2

Tabel 6.2 Hasil Pengujian Parameter *Threshold* nilai *Median*

No	Objek	Jarak	Jarak Hasil Deteksi (cm)			Akurasi %		
			Pagi	Siang	Malam	Pagi	Siang	Malam
1	Box	50 cm	50	43	46	100	86	92
2	Meja	50 cm	50	50	50	100	90	94
3	Orang	50 cm	65	70	32	100	76	64
4	Dinding	50 cm	50	50	50	100	100	100
5	Pintu	50 cm	0	0	0	0	0	0
Rata-rata jarak tiap waktu (%)						**80**	**70,4**	**70**
6	Box	75 cm	75	75	75	100	100	100
7	Meja	75 cm	75	75	78	100	100	96,1
8	Orang	75 cm	90	75	75	83,3	100	100
9	Dinding	75 cm	68	75	75	90,6	100	100
10	Pintu	75 cm	0	75	65	0	100	77.3

Rata-rata jarak tiap waktu (%)						**70,78**	**100**	**94,68**
11	Box	100 cm	100	100	100	100	100	100
12	Meja	100 cm	100	100	105	100	100	95.2
13	Orang	100 cm	100	100	94	100	100	94
14	Dinding	100 cm	100	100	100	100	100	100
15	Pintu	100 cm	0	100	76	0	100	76
Rata-rata jarak tiap waktu (%)						**80**	**100**	**93,04**
16	Box	125 cm	124	125	125	98.4	100	100
17	Meja	125 cm	124	125	125	99.2	100	100
18	Orang	125 cm	0	125	0	0	100	0
19	Dinding	125 cm	125	125	125	100	100	100
20	Pintu	125 cm	125	125	100	100	100	80
Rata-rata jarak tiap waktu (%)						**74,8**	**100**	**76**
Akurasi tiap waktu (%)						**77,39**	**92,6**	**83,43**
Rata-rata akurasi seluruh pengujian (%)						**84,47**		

Dari tabel 6.2 dapat dilihat bahwa dengan menggunakan nilai *threshold median* mendapatkan akurasi sebesar 84,47%. Akurasi terburuk dapat dilihat pada tabel yaitu terjadi pada pagi hari sebesar 77,39%. *Threshold* menggunakan parameter ini memiliki akurasi terbaik ketika siang hari 92,61% dan akurasi yang cukup baik pula pada waktu malam hari yaitu 83,43%



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Gambar 6.2 Hasil pengujian objek *box,* orang dan dinding menggunakan *threshold median*

(a) Pengujian objek box pada pagi hari; (b) Pengujian objek box pada siang hari; (c) Pengujian objek box pada malam hari; (d) Pengujian objek orang pada pagi hari; (e) Pengujian objek orang pada siang hari; (f) Pengujian objek orang pada malam hari; (g) Pengujian objek dinding pada pagi hari; (h) Pengujian objek dinding pada siang hari; (i) Pengujian objek dinding pada malam hari;

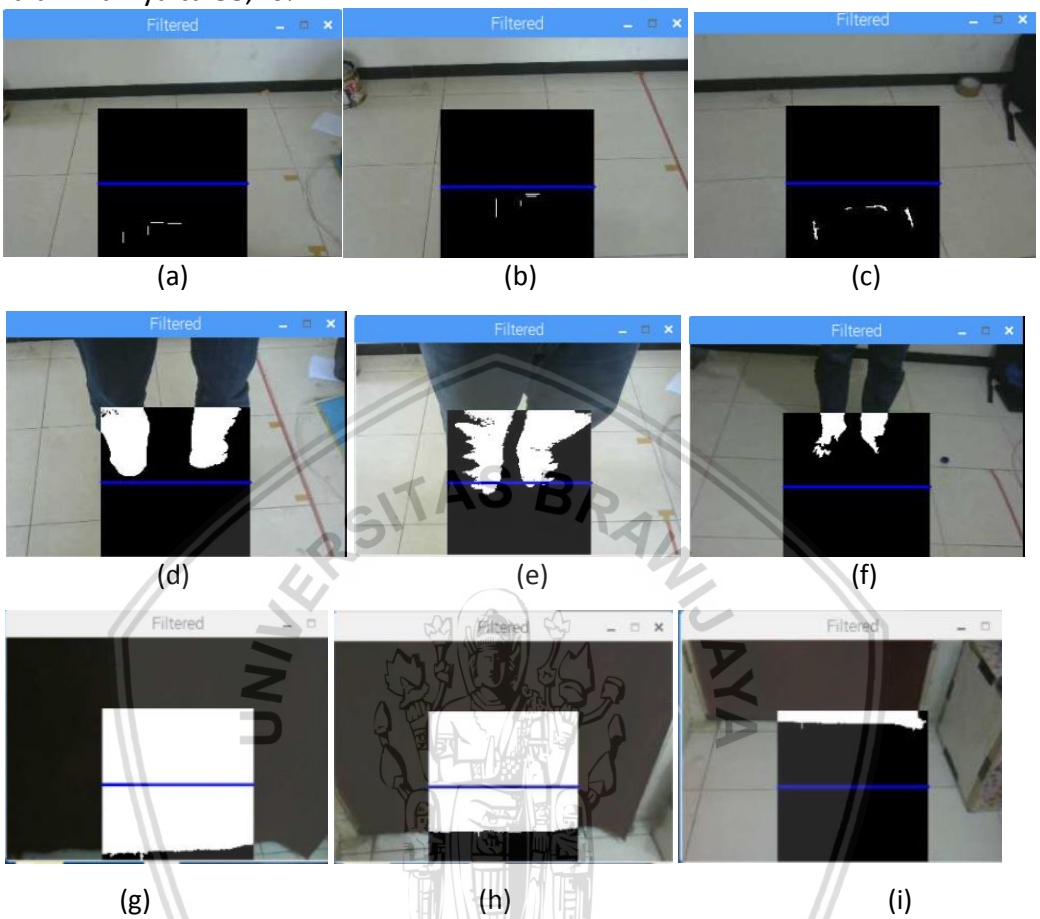
6.1.3.3 *Threshold* dengan Nilai *Min-Max*

Dari data hasil perhitungan BGR yang telah disajikan pada bab 5, maka nilai *threshold* yang digunakan adalah Bmin = 40.0, Bmax = 255.0, Gmin = 49.0 dan Gmax = 255.0, Rmin = 48.0 dan Rmax = 252.0. Hasil pengujian parameter nilai *threshold* dengan nilai *mean* dapat dilihat pada Tabel 6.3

Tabel 6.3 Hasil Pengujian Parameter *Threshold* nilai *Min-Max*

No	Objek	Jarak	Jarak Hasil Deteksi (cm)			Akurasi (%)		
			Pagi	Siang	Malam	Pagi	Siang	Malam
1	Box	50 cm	70	55	62	71.4	90.9	80.6
2	Meja	50 cm	50	0	55	100	0	96
3	Orang	50 cm	85	70	40	58.8	71.4	80
4	Dinding	50 cm	50	50	50	100	100	100
5	Pintu	50 cm	0	0	0	0	0	0
Rata-rata jarak tiap waktu (%)						**64,7**	**42,85**	**69**
6	Box	75 cm	0	75	75	0	100	100
7	Meja	75 cm	0	0	77	0	0	97.4
8	Orang	75 cm	95	90	67	78.9	83.3	89.3
9	Dinding	75 cm	75	75	75	100	100	100
10	Pintu	75 cm	0	0	73	0	0	97.3
Rata-rata jarak tiap waktu (%)						**35,78**	**56,66**	**96,8**
11	Box	100 cm	0	110	100	0	90	100
12	Meja	100 cm	0	0	100	0	99	100
13	Orang	100 cm	120	120	102	83.3	83.3	98
14	Dinding	100 cm	100	100	100	100	100	100
15	Pintu	100 cm	0	100	100	0	100	100
Rata-rata jarak tiap waktu (%)						**36,66**	**94,46**	**99,6**
16	Box	125 cm	0	0	125	0	0	100
17	Meja	125 cm	0	0	125	0	0	100
18	Orang	125 cm	0	0	0	0	0	0
19	Dinding	125 cm	125	125	125	100	100	100
20	Pintu	125 cm	0	125	115	0	100	92
Rata-rata jarak tiap waktu (%)						**20**	**40**	**78,4**
Akurasi tiap waktu (%)						**45,71**	**64,65**	**88,46**
Rata-rata akurasi seluruh pengujian (%)						**62,25**		

Dari tabel 6.3 dapat dilihat bahwa dengan menggunakan nilai *threshold mean* mendapatkan akurasi sebesar 62,25%. Akurasi terburuk dapat dilihat pada tabel yaitu terjadi pada pagi hari 45,71% namun akurasi terbaik didapatkan pada malam hari yaitu 88,46%.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Gambar 6.3 Hasil pengujian objek *box,* orang dan dinding menggunakan *threhold min-max*

(a) Pengujian objek box pada pagi hari; (b) Pengujian objek box pada siang hari; (c) Pengujian objek box pada malam hari; (d) Pengujian objek orang pada pagi hari; (e) Pengujian objek orang pada siang hari; (f) Pengujian objek orang pada malam hari. (g) Pengujian objek dinding pada pagi hari; (h) Pengujian objek dinding pada siang hari; (i) Pengujian objek dinding pada malam hari;

Dari ketiga pengujian parametes *threshold*, dapat dilihat bahwa parameter *threshold* terbaik yaitu dengan menggunakan nilai *mean. Threshold mean* memiliki akurasi sebesar 88,51%. *Threshold median* memiliki akurasi sebesar 84,47% sedangkan *threshold min-max* memiliki akurasi terkecil yaitu 62,25%. Hal ini berati parameter *threshold mean* mampu mendeteksi objek lebih baik dibandingkan dengan parameter *threshold median* dan *threshold min-max*.

6.2 Pengujian dan Analisis Akurasi Sistem dalam Mendeteksi Objek menggunakan *Connected Component Labeling*

6.2.1 Tujuan

Pengujian ini dilakukan untuk mengetahui berapa persen tingkat akurasi sistem dalam mendeteksi objek halangan. Pengujian ini menguji kesesuaian hasil deteksi objek oleh *connected component labeling* dengan output sistem. Pengujian ini dilakukan pada 5 objek berbeda yaitu orang, *box*, tempat sampah, meja dan pintu. Pada pengujian ini, objek berada pada jarak tertentu yaitu 50 cm, 75 cm, 100 cm dan 125 cm di depan pengguna.

6.2.2 Prosedur Pengujian

Prosedur pengujian dilakukan agar pengujian dapat berjalan dengan semestinya, adapun langkah-langkah yang dilakukan adalah sebagai berikut :

1. Melakukan pemasangan *strap belt chest* yang sudah terpasang Raspberry pi, kamera dan buzzer pada dada depan pengguna dengan ketinggian 110 cm.
2. Menghubungkan Raspberry dengan sumber daya.
3. Membuka aplikasi VNC untuk membuka program dan menjalankan program.
4. Memberikan input berupa 5 objek berbeda dan pada jarak yang berbeda-beda (50 cm, 75 cm, 100 cm, dan 125 cm).
5. Mengamati dan mengambil data hasil keluaran sistem.
6. Menentukan tingkat keberhasilan sistem dengan rumus

$$Akurasi = \frac{Total\ data - Data\ tidak\ sesuai}{Total\ Data} \times 100\%$$

6.2.3 Hasil dan Analisis Pengujian

Tabel 6.4 Hasil pengujian deteksi objek pada kondisi pagi, siang, malam

No	Objek	Jarak	Hasil Deteksi			Akurasi (%)
			Pagi	Siang	Malam	
1	Box	50 cm	yes	yes	yes	100
2	Meja	50 cm	yes	yes	yes	100
3	Orang	50 cm	yes	yes	yes	100
4	Dinding	50 cm	yes	yes	yes	100
5	Pintu	50 cm	yes	no	no	66,7
6	Box	75 cm	yes	yes	yes	100
7	Meja	75 cm	yes	yes	yes	100
8	Orang	75 cm	yes	yes	yes	100
9	Dinding	75 cm	yes	yes	yes	100
10	Pintu	75 cm	yes	yes	yes	100

11	Box	100 cm	yes	yes	yes	100
12	Meja	100 cm	yes	yes	yes	100
13	Orang	100 cm	yes	yes	yes	100
14	Dinding	100 cm	yes	yes	yes	100
15	Pintu	100 cm	yes	yes	yes	100
16	Box	125 cm	yes	yes	yes	100
17	Meja	125 cm	yes	yes	yes	100
18	Orang	125 cm	yes	yes	no	33,3
19	Dinding	125 cm	yes	yes	yes	100
20	Pintu	125 cm	yes	no	yes	33,3
Rata-rata akurasi hasil deteksi						**91,66**

Pada tabel 6.4 dapat dilihat bahwa parameter *threshold mean* pada *connected component labeling* dalam mendeteksi objek penghalang memiliki akurasi sebesar 91,66%. Tetapi beberapa objek tidak dapat terdeteksi seperti kulit manusia yang dimana objek tersebut memiliki warna yang mirip dengan warna lantai.

6.3 Pengujian Integrasi *Software* dan *Hardware*

6.3.1 Tujuan

Pengujian ini dilakukan untuk mengetahui berapa persen tingkat akurasi integrasi sistem dalam mendeteksi objek halangan dengan *hardware*. Pengujian ini menguji kesesuaian hasil deteksi objek dengan output sistem. Pengujian ini dilakukan pada 5 objek berbeda yaitu orang, *box*, tempat sampah, meja dan pintu. Pada pengujian ini, objek berada pada jarak tertentu yaitu 50 cm, 75 cm, 100 cm dan 125 cm di depan pengguna.

6.3.2 Prosedur Pengujian

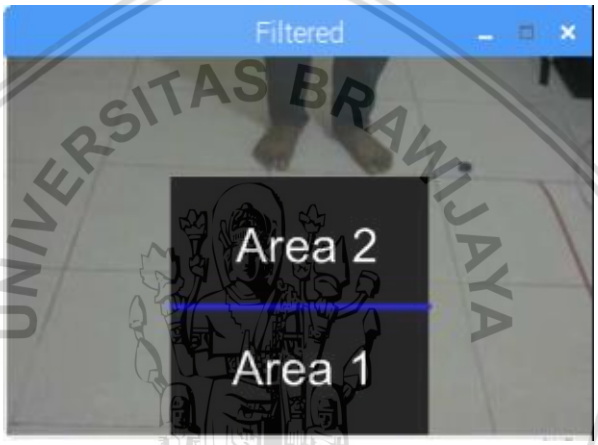
Prosedur pengujian dilakukan agar pengujian dapat berjalan dengan semestinya, adapun langkah-langkah yang dilakukan adalah sebagai berikut :

1. Melakukan pemasangan *strap belt chest* yang sudah terpasang Raspberry pi, kamera dan buzzer pada dada depan pengguna dengan ketinggian 110 cm.
2. Menghubungkan Raspberry dengan sumber daya.
3. Membuka aplikasi VNC untuk membuka program dan menjalankan program.
4. Memberikan input berupa 5 objek berbeda dan pada jarak yang berbeda-beda (50 cm, 75 cm, 100 cm, dan 125 cm).
5. Mengamati dan mengambil data hasil keluaran sistem.

6.3.3 Pelaksanaan Pengujian

Pengujian ini dilakukan dengan 5 objek berbeda yaitu box, meja, orang, dinding dan pintu dengan jarak masing-masing 50 cm, 75 cm, 100 cm dan 125 cm pada waktu pagi, siang dan malam. Untuk untuk mendapatkan presentase yaitu dengan menghitung total data dikurangi data yang tidak sesuai kemudian dibagi dengan total data dan dikalikan 100%

Pada Gambar 6.4 merupakan tampilan hasil sistem yang telah dicropping dengan ukuran 150 x 150 piksel. Dalam ukuran tersebut dibagi dua menajdi Area 1 dan Area 2. Ketika objek terlihat pada area 1 maka dapat direpresentasikan bahwa objek telah dekat dengan pengguna sehingga *buzzer* akan berbunyi keras. Ketika objek terlihat pada area 2 saja maka direpresentasikan bahwa objek masih tidak begitu dekat dengan pengguna maka *buzzer* akan berbunyi lemah. Namun ketika objek terlihat pada area 1 dan 2 maka *buzzer* akan berbunyi keras.



Gambar 6.4 Area 1 dan area 2 pada pengujian integrasi sistem dengan *hardware*

6.3.4 Hasil dan Analisis Pengujian

Tabel 6.5 Hasil pengujian Integrasi Sistem Software dan hardware pada Waktu Pagi Hari

No	Objek	Jarak (cm)	Deteksi		Bunyi Buzzer		Akurasi (sesuai/tidak sesuai)
			Area 1	Area 2	Keras	Lemah	
1	Box	50 cm	yes	-	-	yes	Sesuai
2	Meja	50 cm	yes	yes	yes	-	Sesuai
3	Orang	50 cm	yes	yes	Yes	-	Sesuai
4	Dinding	50 cm	yes	-	-	yes	Sesuai
5	Pintu	50 cm	yes	yes	yes	-	Sesuai
6	Box	75 cm	yes	yes	yes	-	Sesuai
7	Meja	75 cm	yes	yes	yes	-	Sesuai
8	Orang	75 cm	-	yes	yes	-	Sesuai
9	Dinding	75 cm	yes	-	-	yes	Sesuai
10	Pintu	75 cm	yes	yes	yes	-	Sesuai

11	Box	100 cm	yes	yes	yes	-	Sesuai
12	Meja	100 cm	-	yes	-	yes	Sesuai
13	Orang	100 cm	-	yes	-	yes	Sesuai
14	Dinding	100 cm	-	yes	-	yes	Sesuai
15	Pintu	100 cm	yes	yes	yes	-	Sesuai
16	Box	125 cm	-	yes	-	yes	Sesuai
17	Meja	125 cm	-	yes	-	yes	Sesuai
18	Orang	125 cm	-	yes	-	yes	Sesuai
19	Dinding	125 cm	-	yes	-	yes	Sesuai
20	Pintu	125 cm	-	yes	-	yes	Sesuai
Akurasi integrasi software dan Hardware pada waktu pagi (%)							100%

Pada tabel 6.5 dapat dilihat bahwa akurasi dari hasil pengujian integrasi sistem software dan software pada siang hari yaitu 100%

Tabel 6.6 Hasil Pengujian Integrasi Sistem *Software* dan *Hardware* pada waktu Siang

No	Objek	Jarak (cm)	Deteksi		Bunyi Buzzer		Akurasi
			Area 1	Area 2	Keras	Lemah	
1	Box	50 cm	yes	-	-	yes	Sesuai
2	Meja	50 cm	yes	yes	yes	-	Sesuai
3	Orang	50 cm	yes	yes	yes	-	Sesuai
4	Dinding	50 cm	yes	-	-	yes	Sesuai
5	Pintu	50 cm	-	-	-	-	Sesuai
6	Box	75 cm	yes	yes	yes	-	Sesuai
7	Meja	75 cm	yes	yes	yes	-	Sesuai
8	Orang	75 cm	-	yes	-	yes	Sesuai
9	Dinding	75 cm	yes	-	-	yes	Sesuai
10	Pintu	75 cm	yes	yes	yes	-	Sesuai
11	Box	100 cm	-	yes	-	yes	Sesuai
12	Meja	100 cm	-	yes	-	yes	Sesuai
13	Orang	100 cm	-	yes	-	yes	Sesuai
14	Dinding	100 cm	-	yes	-	yes	Sesuai
15	Pintu	100 cm	-	yes	-	yes	Sesuai
16	Box	125 cm	-	yes	-	yes	Sesuai
17	Meja	125 cm	-	yes	-	yes	Sesuai
18	Orang	125 cm	-	yes	-	yes	Sesuai
19	Dinding	125 cm	-	yes	-	yes	Sesuai
20	Pintu	125 cm	-	-	-	-	Sesuai
Akurasi integrasi software dan Hardware pada waktu pagi (%)							100%

Pada tabel 6.6 dapat dilihat bahwa akurasi dari hasil pengujian integrasi sistem software dan software pada pagi hari yaitu 100%

Tabel 6.7 Hasil Pengujian Sistem Integrasi *Software* dan *Hardware* pada Waktu Malam

No	Objek	Jarak (cm)	Deteksi		Bunyi Buzzer		Akurasi
			Area 1	Area 2	Keras	Lemah	
1	Box	50 cm	yes	-	-	yes	Sesuai
2	Meja	50 cm	yes	yes	yes	-	Sesuai
3	Orang	50 cm	yes	yes	yes	-	Sesuai
4	Dinding	50 cm	yes	-	-	yes	Sesuai
5	Pintu	50 cm	-	-	-	-	Sesuai
6	Box	75 cm	yes	yes	yes	-	Sesuai
7	Meja	75 cm	yes	yes	yes	-	Sesuai
8	Orang	75 cm	yes	yes	yes	-	Sesuai
9	Dinding	75 cm	yes	-	-	yes	Sesuai
10	Pintu	75 cm	yes	yes	yes	-	Sesuai
11	Box	100 cm	-	yes	-	yes	Sesuai
12	Meja	100 cm	-	yes	-	yes	Sesuai
13	Orang	100 cm	-	yes	-	yes	Sesuai
14	Dinding	100 cm	-	yes	-	yes	Sesuai
15	Pintu	100 cm	yes	yes	yes	-	Sesuai
16	Box	125 cm	-	yes	-	yes	Sesuai
17	Meja	125 cm	-	yes	-	yes	Sesuai
18	Orang	125 cm	-	-	yes	-	Tidak Sesuai
19	Dinding	125 cm	-	yes	-	yes	Sesuai
20	Pintu	125 cm	-	yes	-	yes	Sesuai
Akurasi integrasi software dan Hardware pada waktu pagi (%)							95%

Pada Tabel 6.7 data tidak sesuai terdapat pada no.18 yaitu objek orang pada jarak 125 cm. Sistem pada software telah mendeteksi noise yang mengakibatkan buzzer berbunyi namun sistem tidak mendeteksi objek maka akurasi dari hasil pengujian integrasi sistem software dan software pada malam hari yaitu 95%.

Dari hasil pengujian integrasi sistem *software* dan *hardware* pada waktu pagi siang dan malam memiliki rata-rata akurasi yaitu 98,33%

6.4 Pengujian dan Analisis Waktu Komputasi Sistem

6.4.1 Tujuan

Pengujian ini dilakukan untuk mengetahui berapa lama waktu yang dibutuhkan sistem untuk menyelesaikan proses hingga mendapatkan hasil keluaran.

6.4.2 Prosedur Pengujian

Prosedur pengujian dilakukan agar pengujian dapat berjalan dengan semestinya, adapun langkah-langkah yang dilakukan adalah sebagai berikut :

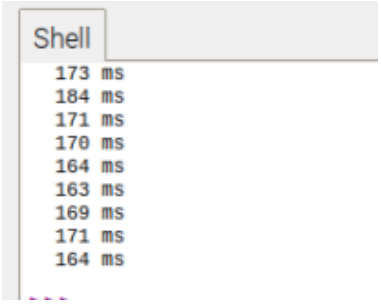
1. Melakukan pemasangan *strap belt chest* yang sudah terpasang Raspberry pi, kamera dan buzzer pada dada depan pengguna dengan ketinggian 110 cm.
2. Menghubungkan Raspberry dengan sumber daya.
3. Membuka aplikasi VNC untuk membuka program dan menjalankan program.
4. Memberikan input berupa 5 objek berbeda dan pada jarak yang berbeda-beda (50 cm, 75 cm, 100 cm, dan 125 cm).
5. Mengamati dan mengambil data hasil keluaran sistem. Setelah mendapat hasil waktu komputasi untuk deteksi objek, maka langkah selanjutnya adalah menghitung rata-rata waktu komputasi.

6.4.3 Hasil dan Analisis Pengujian

Tabel 6. 8 Hasil Pengujian Waktu Komputasi

No	Objek	Jarak	Waktu Komputasi (ms)
1	Box	50 cm	167
2	Meja	50 cm	166
3	Orang	50 cm	161
4	Dinding	50 cm	162
5	Pintu	50 cm	163
Rata-rata waktu komputasi pada jarak 50 cm			**163,8**
6	Box	75 cm	162
7	Meja	75 cm	170
8	Orang	75 cm	158
9	Dinding	75 cm	180
10	Pintu	75 cm	163
Rata-rata waktu komputasi pada jarak 75 cm			**166,6**
11	Box	100 cm	164
12	Meja	100 cm	161
13	Orang	100 cm	172
14	Dinding	100 cm	177
15	Pintu	100 cm	181
Rata-rata waktu komputasi pada jarak 100 cm			**171**
16	Box	125 cm	159
17	Meja	125 cm	176
18	Orang	125 cm	167
19	Dinding	125 cm	167
20	Pintu	125 cm	157
Rata-rata waktu komputasi pada jarak 125 cm			**167,2**
Rata-rata waktu komputasi			**166,15 ms**

Dari tabel 6.8 diatas merupakan hasil pengujian waktu komputasi deteksi objek penghalang. Dari 20 kali pengujian yang telah dilakukan, mendapatkan nilai rata-rata waktu komputasi yaitu 166,15 ms.



Gambar 6. 5 *Output waktu komputasi*

BAB 7 PENUTUP

7.1 Kesimpulan

Berdasarkan proses penelitian yang sudah dilakukan, rumusan masalah yang telah dibuat sebelumnya dan hasil penelitian yang ada, maka diapat disimpulkan bahwa :

1. Sistem dapat mendeteksi objek penghalang dengan menggunakan beberapa parameter *threshold* yaitu parameter nilai *thresold mean, median,* dan *min-max* pada jarak yang berbeda beda menghasilkan nilai akurasi yang berbeda-beda. Dari ketiga batas *threshold* tersebut, th*reshold mean* memiliki akurasi terbaik yaitu 88,51%. Sedangkan akurasi terburuk yaitu *threshold min-max* yang memiliki akurasi sebesar 66,25% sedangkan akurasi *threshold median* yaitu 84,47%
2. Akurasi sistem dalam mendeteksi objek penghalang ini dapat dilihat pada hasil pengujian. Pengujian dilakukan di dalam ruangan dengan pada waktu pagi siang dan malam memiliki akurasi sebesar 91,66% tetapi dalam sistem ini masih memiliki kekurangan yaitu masih mengaggap cahaya pantulan, bayangan dan objek yang berwarna mirip dengan warna lantai akan tidak dapat terdeteksi. Hal ini dikarenakan sistem masih bergantung pada komponen RGB dari citra.
3. Akurasi integrasi sistem *software* dengan *hardware* yang diujikan pada waktu pagi, siang dan malam yaitu sebesar 98,33% hal ini dipengaruhi oleh *noise* seperti cahaya lampu yang memantul pada lantai atau bayangan pada lantai saat sistem digunakan.
4. Waktu komputasi pada pengujian deteksi objek penghalang ini memiliki nilai yang berbeda-beda. Rata-rata waktu komputasi pada sistem deteksi objek penghalang ini yaitu 166,15 ms

7.2 Saran

1. Untuk mendeteksi objek penghalang hendaknya menggunakan atau menambahkan metode yang lain untuk hasil deteksi objek yang lebih akurat
2. Untuk penelitian selanjutnya hendaknya dapat diimplementasikan tidak hanya di dalam ruangan saja, namun di luar ruangan
3. Menggunakan kamera jenis lain untuk dapat mendapatkan akurasi yang lebih baik dalam meng*capture* objek.

DAFTAR PUSTAKA

Al Kadafi, A. J., 2017. *Deteksi Objek Penghalang Secara Real-Time Berbasis Mobile bagi Penyandang Tunanetra Menggunakan Analisis Blob.* Malang: s.n.

Al Kadafi, A. J. & Utaminungrum, F., 2017. Deteksi Objek Penghalang Secara Real-Time Berbasis Mobile Bagi Penyandang Tunanetra Menggunakan Analisis Blob. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer,* Volume 2, pp. 423 - 432.

Amri, S., Suyono, H. & Setyawati, O., 2014. Perbandingan Metode CF dan k-NN untuk Identifikasi Warna pada Robot Soccer. *Jurnal EECCIS,* Volume 8.

Ardhianto, E., Hadikurniawati, W. & Budiarso, Z., 2013. Implementasi Metode Image Subtracting dan Metode Regionprops untuk Mendeteksi Jumlah Objek Berwarna RGB pada File Video. *Jurnal Teknologi Informasi DINAMIK,* Volume 18, p. 95.

Budisanjaya, I. P. G. & Kumara, I. N. S., 2013. Perangkat Lunak Pengolahan Citra Untuk Segmentasi dan Cropping Daun Sawi Hijau. *Prosiding Conference on Smart-Green Technology in Electrical and Information System,* p. 217.

Cahyana, F. M., Julius, M. & Setiawan, R. A., 2014. Implementasi Invers Kinematics Pada Sistem. *Jurnal Mahasiswa Elektro Universitas Brawijaya,* p. 1.

Fanani, A., Prima, P. & Hidayat, M. M., 2012. Local Thresholding Berdasarkan Bentuk. *Jurnal Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember,* Volume 10, p. 27.

Firdausy, K., Saudi, Y. & Sutikno, T., 2007. Deteksi Api Real-Time dengan Metode Thresholding Rerata RGB. *TELKOMNIKA,* Volume 5, p. 127.

Harimukhti, M. T. & Dewi, K. S., 2014. Ekplorasi Kesejahteraan Psikologis Individu Dewasa Awal Penyandang Tunanetra. *Jurnal Psikologi Undip,* Volume 13, pp. 64-77.

Helmirawan, 2012. Rancang Bangun Dan Analisis Sistem Pemantau Lalu Lintas Menggunakan Opencv Dengan Menggunakan Algoritma Canny Dan Blob Detection. *Skripsi Universitas Indonesia.*

Hendry, J., 2010. *SCRIBD.* [Online] Available at: https://id.scribd.com/document/56440168/Region-Clustering-Dengan-Mengunakan-Connected-Component-Labeling-Pada-Citra-Digital [Diakses 10 August 2018].

HobbyTronics, 2018. *Hobby Tronics.* [Online] Available at: http://www.hobbytronics.co.uk/5v-buzzer [Diakses 9 7 2018].

Irianto, K. D., 2010. Pendeteksi Gerak Berbasis Kamera Menggunakan Opencv Pada Ruangan. *Jurnal KomuniTi,* Volume 2.

Isharwati, 2008. Mengenali Penyandang Tunanetra dan Intervensi Pendidikannya.

Kadmiri, z. E., Kadmiri, O. E. & Masmoudi, L., 2012. Depth Estimation For Mobile Robot Using Single. *Journal of Theoretical and Applied Information Technology,* Volume 44, p. 30.

Kusumanto, R. & Tompunu, A. N., 2011. Pengolahan Citra Digital untuk Mendeteksi Obyek Menggunakan pengolahan Warna Model Normalisasi RGB. *Seminar Nasional Teknologi Informasi & Komunikasi Terapan,* p. 1.

Kusumanto, R., Tompunu, A. N. & Pambudi, W. S., 2011. Klasifikasi Warna Menggunakan Pengolahan Model Warna HSV. *Jurnal Ilmiah Elite Elektro,* Volume 2, p. 1.

Logitech, 2018. *Logitech.* [Online]
Available at: https://www.logitech.com/en-us/product/hd-webcam-c310
[Diakses 10 8 2018].

MATT, 2016. *Introducing the Raspberry Pi 3 Model B Single Board Computer.* [Online]
Available at: https://www.raspberrypi-spy.co.uk/2016/02/introducing-the-raspberry-pi-3-model-b/
[Diakses 15 July 2018].

Mirul, K., 2017. *It's Sience.* [Online]
Available at: http://k-sience.blogspot.com/2017/06/deteksi-jumlah-objek-dengan-metode.html
[Diakses 10 August 2018].

Murinto & Harjoko, A., 2009. Segmantasi Citra Menggunakan Watershed dan Intensitas Filtering Sebagai Pre Processing. *Seminar Nasional informatika,* pp. A-43.

N, S. & S, V., 2016. Image Segmentation By Using Thresholding. *Computer Science & Engineering: An International Journal (CSEIJ),* Volume 6, p. 2.

OpenCV, t., 2018. *OpenCV.* [Online]
Available at: https://opencv.org/
[Diakses 15 July 2018].

Pramana, C. J., 2015. Implementasi Metode Thresholding dan Metode Regionprops untuk Mendeteksi Marka Jalan secara Live Video. *Skripsi Fakultas Ilmu Komputer Universitas Dian Nuswantoro,* p. 2.

Putranto, B. Y. B., Hapsari, W. & Wijana, K., 2012. Segmentasi Warna Citra Dengan Deteksi Warna HSV Untuk Mendeteksi Objek. *Jurnal Informatika Fakultas Teknik Program Studi Teknik Informatika.*

Raspberry Pi Foundation, 2016. *Raspberry Pi.* [Online]
Available at: https://www.raspberrypi.org/products/raspberry-pi-3-

model-b/
[Diakses 16 8 2018].

Rizki, A. et al., 2010. Connected Component Analysis Sebagai Metode Pencarian Karakter Plat dalam Sistem Pengenalan Plat Nomor Kendaraan. *Proc. of 11th Seminar on Intelligent Technology & Its Application (SITIA 2010),,* pp. 300-305.

Robotix, 2018. *Robotix - Technology Robotix Society.* [Online] Available at: https://2018.robotix.in/tutorial/imageprocessing/imagetypes/ [Diakses 31 8 2018].

Salem, O. G., 2014. *Code Project.* [Online] Available at: https://www.codeproject.com/Articles/336915/Connected-Component-Labeling-Algorithm [Diakses 10 August 2018].

Sardegna, J. S. S. R. A. &. S. S., 2002. *The encyclopedia of blindess and vision impairement.* New York: Fact On File.

Schwenk, K. H. F., 2015. *Connected Component Labelling Algorithm for very complex and High Resolution Images on an FPGA platform.* s.l.:s.n.

Silvia P, M., 2017. *World Health Organiza□on.* [Online] Available at: http://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment [Diakses 27 May 2018].

Soemantri, S., 2012. *Psikologi anak luar biasa.* Bandung: Refika Aditama.

Sulistiyowati, R. & Febriantoro, D. D., 2012. Perancangan Prototype Sistem Kontrol dan Monitoring Pembatas Daya Listrik Berbasis Mikrikntroler. *Jurnal IPTEK,* Volume 16, p. 26.

Wahyudi, D. A. & Kartowisastro, I. H., 2011. Menghitung Kecepatan Menggunakan Computer Vision. *Jurnal Computer Engineering Binus University,* Volume 19, p. 101.

Wardani, I. K., 2012. *academia edu.* [Online] Available at: https://www.academia.edu/11958478/strategi_presentase_diri_mahasiswa_tunanetra [Diakses 21 Maret 2018].